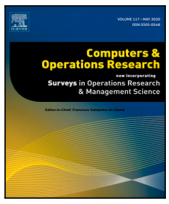
Computers & Operations Research 182 (2025) 107144

Contents lists available at ScienceDirect

Computers and Operations Research

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



Scheduling with assignable due dates, two competing agents and late work related criteria

Shi-Sheng Li [a,*], Yao-Wen Sang [b], Ren-Xia Chen [a], Malgorzata Sterna [c], Tamas Kis [d]

[a] School of Mathematics and Information Science, Zhongyuan University of Technology, Zhengzhou 450007, People's Republic of China
[b] College of Economics and Management, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, People's Republic of China
[c] Institute of Computing Science, Poznan University of Technology, Piotrowo 2, Poznan 60-965, Poland
[d] HUN-REN Institute for Computer Science and Control, Kende utca 13-17, Budapest, 1111, Hungary

ARTICLE INFO

Keywords:
Scheduling
Single machine
Two-agent
Assignable due dates
Late work

ABSTRACT

This paper investigates several scheduling problems on a single machine that exhibit three key features: (i) assignable due dates are considered, meaning that a predefined set of due dates is provided, and these due dates can be assigned to the jobs independently and arbitrarily; (ii) two competing agents are involved, with jobs divided into two disjoint sets, each owned by one agent; and (iii) each agent's objective is to minimize a specific late work-related criterion. Specifically, the goal is to determine a due date assignment and a feasible schedule that minimizes the weighted sum of the selected scheduling criteria for both agents. For the first time, we integrate two-agent scheduling, assignable due dates, and late work-related criteria. We develop pseudo-polynomial-time dynamic programming algorithms to address problems where one agent's criterion is the total late work, while the other agent's criterion is either the total late work, total tardiness, or weighted number of tardy jobs. Moreover, we propose polynomial-time algorithms to tackle various special cases of these related problems.

1. Introduction

As production scale and complexity increase, effectively managing and optimizing all aspects of the production process has become a significant challenge for firms. An important part of this process is to minimize the due-date-related scheduling measures (Atsmony et al., 2023; Prata et al., 2025). In the traditional scheduling involving due-date-related criteria (Blazewicz et al., 2019), each job is related to a fixed due date. However, in modern Just-In-Time (JIT) manufacturing setting, the due date of a job d_j is usually determined according to a specific due date assignment rule and its value contributes to the overall scheduling objective function (Gordon et al., 2002; Rolim and Nagano, 2020). The due date assignment rules commonly used in the literature are based on the following strategies: (i) the CON approach (all jobs are assigned a common due date, i.e., $d_j = d$); (ii) the SLK approach (the due date of each job is defined as its processing time p_j plus an equal slack, i.e., $d_j = p_j + d$); (iii) the PPW approach (the due dates are assigned according to the processing-plus-waiting-time rule so that $d_j = kp_j + d$, where $d \geq 0$ and $k \geq 0$ are both decision variables); (iv) the DIF approach (the due dates are assigned without any restrictions). Alternatively, Hall (1986) introduced the concept of *generalized due dates* (GDD), where a set of due dates is given in advance and the kth smallest due date is assigned to the kth completed job in a given schedule. Qi et al. (2002) introduced another concept of *assignable due dates* (ADD), where a predefined set of due dates is given and these due dates can be assigned to the jobs independently and arbitrarily.

At the same time, with the development of economy and market expansion, manufacturers increasingly adopt collaborative strategies such as partnerships and joint production to deal with market competition. This has shifted the focus of scheduling towards the effective utilization of machine resources and meeting the diverse demands of individual customers. Consequently, multi-agent scheduling has emerged as a hot topic in the research community. Generally, in a multi-agent scheduling model, two or more agents need to use shared production resources and equipment to complete their respective set of jobs, and each agent desires to minimize or maximize its own scheduling criterion. This scenario arises in numerous production and manufacturing environments, including preventive maintenance planning (Leung et al., 2010; Pal et al., 2023), supply chain scheduling (Agnetis et al., 2014; Xu et al., 2024), batch scheduling (Kovalyov et al., 2015; Fan et al., 2023), and resource-constrained project scheduling (Geng and Yuan, 2023; Rodriguez-Ballesteros et al., 2024), etc.

* Corresponding author.
E-mail address: shishengli96@163.com (S.-S. Li).

https://doi.org/10.1016/j.cor.2025.107144
Received 19 September 2024; Received in revised form 9 May 2025; Accepted 13 May 2025
Available online 28 May 2025

In this paper, we address scheduling problems that involve assignable due dates (ADD), two competing agents and late work related measures. Both two-agent scheduling and ADD models find numerous practical applications, which appear among others in airline industry (Yin et al., 2012), integrated production and batch delivery (Agnetis et al., 2017) and manufacturing (Justkowiak et al., 2023). Similarly, late work related criteria are applicable in control systems (Blazewicz, 1984), agriculture (Blazewicz et al., 2004), flexible manufacturing (Sterna, 2007), software development (Sterna, 2011), and logistics industry (Chen et al., 2023). In the presented studies, we combine for the first time these three mentioned models opening a new research stream. Consider an integrated production and transportation scenario in a manufacturing company, where various orders (jobs) with different requirements come from two distinct clients (agents). Each client specifies a set of delivery dates (due dates) for their respective orders. Initially, these orders are processed on the company's production line before being dispatched to their designated destinations. Following production, the orders are eligible for pickup and delivery on specific dates, which are drawn from each client's respective delivery date set. Importantly, each order must be assigned a unique delivery date. Timely manufacturing and transportation of orders at or before their assigned delivery dates incurs no additional costs. However, failing to meet these promised delivery dates, whether partially or fully, results in penalties such as storage fees, customer dissatisfaction, or damage to the company's reputation. From each client's perspective, the primary objective is to minimize its own due-date-related measure, such as the weighted number of tardy orders, total amount of processing beyond delivery dates (i.e., late work), or total tardiness. Conversely, the manufacturer aims to devise a scheduling scheme and a delivery date assignment strategy that minimizes the weighted sum of the due-date-related measures for both clients. This scenario can thus be conceptualized as a scheduling model for a single machine involving two competing agents, due date assignment and late work related measures.

The model under investigation is intimately tied to the streams of ADD scheduling, late work scheduling and two-agent scheduling in the literature.

The ADD scheduling model was first introduced by Qi et al. (2002). They present two practical applications for the model: one involving a manufacturing scenario where the challenge is scheduling product deliveries at specific times after job completion, and the other relating to the scheduling of airline pilot training courses to ensure that pilots are ready for duty at specific times. Qi et al. (2002) provide a detailed complexity analysis for a range of single-machine scheduling problems. Specifically, their main results include: (i) when the objective is to minimize the total tardiness, maximum lateness and number of tardy jobs, efficient polynomial-time algorithms are developed; (ii) when the objective is to minimize the weighted number of tardy jobs, binary NP-hardness is proved and a pseudo-polynomial-time dynamic programming (DP) algorithm is designed; (iii) when the objective is to minimize the total earliness-tardiness, a proof of NP-hardness is constructed. Additionally, Gao and Yuan (2015) demonstrate that the total earliness-tardiness minimization problem is unary NP-hard.

The late work related criteria, due to their wide practical applications, have attracted considerable attention in recent years. Various scheduling models with additional parameters and constraints have been investigated in the literature (Sterna, 2011, 2021). Yin et al. (2016) study a single-machine scheduling problem with the goal of minimizing the total late work, where the machine undergoes a maintenance period during which it cannot perform any job. They design two pseudo-polynomial-time DP algorithms and a fully polynomial time approximation scheme (FPTAS). Mosheiov et al. (2021) demonstrate that the single-machine GDD scheduling to minimize the total late work is polynomially solvable. They extend the problem by allowing job rejection and considering a maintenance period. Li and Chen (2022) continue the work of Yin et al. (2016) by considering multiple maintenance periods and the total weighted late work criterion. They analyze the computational complexity of the preemptive, resumable and non-resumable variants of the problem. Shabtay et al. (2022) investigate the problem of minimizing the overall weighted early and late work in the setting of assignable common due date/window on a single machine. Shabtay (2023) studies single-machine scheduling with DIF assignment to minimize the overall weighted early and late work, while an assignable due date of each job is constrained by a job-specific upper bound. Furthermore, Shabtay (2023) offers a fresh perspective on minimizing the total late work by analyzing its parameterized complexity with respect to various parameters, including the number of distinct due dates, the number of distinct processing times, the maximum due date and the maximum processing time. Sang et al. (2023) address the CON and DIF assignment models on a single machine, seeking to minimize the overall weighted late work and lead time penalty, where the latter is defined as the duration by which an assigned due date exceeds a fixed lead time. Chen et al. (2024) investigate the proportionate flow shop scheduling with job rejection, in which the objective is to minimize the sum of the total weighted late work for accepted jobs and rejection cost for rejected ones. To solve this problem, they design a pseudo-polynomial-time algorithm and an FPTAS. Sang et al. (2025) address scheduling with the total weighted late work and job rejection on a single machine. They devise pseudo-polynomial-time algorithms to solve the Pareto version and two restricted versions. Phosavanh and Oron (2025) study the problem of minimizing the total late work in the context of step-learning, where step learning refers to the scenario where a job's processing time decreases if it is started on or after its individual learning date. In terms of parallel-machine scheduling, the literature mainly focuses on the maximization of the total (weighted) early work with a common due date, where early work mirrors the part of the job scheduled before its due date (Sterna, 2021). For example, for the online or semi-online cases, Chen et al. (2021) and Jiang et al. (2024) derive lower bounds and analyze the performance of online algorithms by providing upper bounds on their competitive ratios. Sterna and Czerniachowska (2017), Chen et al. (2020, 2022b), Choi et al. (2021) and Li (2024) study the offline case by designing and analyzing various exact and approximation algorithms.

There are also some works that combine late work criteria with ADD scheduling in the single machine setting. Mosheiov et al. (2021) demonstrate that the total late work minimization problem is NP-hard, but no solution algorithm is presented. Justkowiak et al. (2023) and Chen et al. (2023) construct pseudo-polynomial-time DP algorithms and FPTASes to solve the problem proposed by Mosheiov et al. (2021), respectively. Moreover, Justkowiak et al. (2023) develop an $O(n \log n)$-time algorithm for the maximum late work minimization problem, where n is the number of jobs. Chen et al. (2023) demonstrate that the total weighted late work minimization problem is unary NP-hard, and propose an $O(n \log n)$-time algorithm for the case where all jobs have a common processing time. Chen et al. (2025) explore the scheduling problem of minimizing the total weighted late work with assignable due dates and job preemption. They show that the problem is NP-hard and provide a pseudo-polynomial-time algorithm.

Another issue that is closely related to our study is the two-agent scheduling, which is initiated by the two pioneering papers presented by Baker and Smith (2003) and Agnetis et al. (2004). In their innovative investigations, the scheduling criteria involved are the makespan, maximum lateness, total weighted completion time, and number of tardy jobs. For different combinations of scheduling criteria employed by the two agents, Baker and Smith (2003) explore the problem from the approach tailored to the weighted sum variant, whereas Agnetis et al. (2004) address the problem from the perspectives of constrained version and Pareto version. More broadly speaking, the field of scheduling involving two or more agents has been the subject of extensive research, a universal framework and taxonomy concerning computational complexity and solution methods/procedures for the existing

literature up to 2014 are presented in the book by Agnetis et al. (2014) as well as in the review paper by Perez-Gonzalez and Framinan (2014). Agent scheduling has numerous applications in various real-life fields, witnessed by the diverse models explored in the literature. The most recent research on two-agent scheduling models considers among others: batch scheduling (e.g., He et al., 2022, Fan et al., 2023), due date assignment scheduling (e.g., Yin et al., 2020; Zhang et al., 2023; Li et al., 2024), scheduling with optional job rejection (e.g., Oron, 2021; Freud and Mosheiov, 2021), scheduling with release dates and preemption (e.g., Li and Chen, 2023; Gu et al., 2024), scheduling with a rate-modifying activity (e.g., Pal et al., 2023; Phosavanh and Oron, 2024), and shop scheduling (e.g., Liu et al., 2024; Wang et al., 2024), etc.

The criteria related to late work have also been explored in the context of two-agent scheduling within a single-machine framework. Wang et al. (2017) investigate the problem of minimizing the total late work for the first agent, while imposing an upper bound on the maximum lateness for the second agent. They devise two pseudo-polynomial-time DP algorithms and a branch-and-bound algorithm to tackle this issue. Zhang and Yuan (2019) demonstrate that the constrained version of the scheduling problem is binary NP-hard when one criterion is the makespan and the other is the total late work. Li and Yuan (2020) address a scheduling problem involving m competing agents, where each agent aims to minimize its own total weighted late work. They reveal that the constrained version of the unweighted problem is unary NP-hard for arbitrary m, and they provide a pseudo-polynomial-time algorithm along with an approximate Pareto-optimal frontier for the Pareto version when m is fixed. Zhang et al. (2021) consider a three-agent scheduling scenario where the criteria for evaluation are the weighted number of tardy jobs, total weighted completion time and total weighted late work, respectively. Chen et al. (2022a) focus on Pareto-scheduling with two agents, where the jobs of each agent share a common processing time. They establish that the problems are binary NP-hard when the first agent's criterion is the total late work, and the second agent's criterion is either the total weighted completion time, weighted number of tardy jobs, total late work, or total tardiness.

Though both ADD scheduling and two-agent scheduling have received increasing research interest, to our knowledge, the only studies that investigate scheduling involving both ADD and two agents are Yin et al. (2012) and Wang et al. (2015). In their studies, the scheduling criteria involved are the maximum lateness, total tardiness and weighted number of tardy jobs. For different combinations of scheduling criteria utilized by the two agents, Yin et al. (2012) investigate the weighted sum version, while Wang et al. (2015) deal with the constrained variant. They provide polynomial-time or pseudo-polynomial-time algorithms for these problems. However, the late work criterion has not been explored among all these studies.

Taking the above issues into consideration, our research mainly focuses on the innovative single-machine ADD scheduling problem, involving two competing agents and late work related measures. This problem represents the first attempt to integrate these diverse research streams. We begin our investigation with the basic single-machine problem, aiming to pave the way for future research on more complex scheduling models. Determining the complexity status of such basic problems, i.e., proving their polynomial time solvability or NP-hardness, is necessary to find the borderline between "easy" and "intractable" cases (Garey and Johnson, 1979). If the single-machine problem is "intractable", i.e., NP-hard, then all more complex models are also NP-hard and require exponential methods to be solved optimally. Otherwise, if they are "easy", i.e. polynomially solvable, the more complex models still require dedicated complexity studies, but the results obtained for a single-machine model provide invaluable hints for them. Particularly, the properties valid for single-machine scheduling can also be useful for constructing algorithms and proofs for multi-machine models, and algorithms that are optimal for single-machine can serve as heuristics for intractable multi-machine cases. Scheduling models encountered in practice are usually much more complex than the one studied in the paper. However, the theoretical studies presented here are necessary and crucial, as they establish the groundwork for exploring and understanding more advanced scheduling models within this new research stream.

The paper is outlined below. In Section 2, we provide a detailed description of the problem under study, along with some necessary notations. In Section 3, we first analyze structural properties of optimal solutions, then propose pseudo-polynomial-time DP algorithms for the general problems, and finally evaluate their performance through extensive numerical experiments. In Section 4, we design polynomial-time algorithms to tackle various special cases of the related problems. In Section 5, we summarize the paper and outline some topics for future research.

2. Problem description and notations

The problem deals with two competing agents, named agents A and B, respectively. Let $X \in \{A, B\}$. Agent X needs to schedule set $\mathcal{J}^X = \{J_1^X, J_2^X, \ldots, J_{n_X}^X\}$ of X-jobs, where $\mathcal{J}^A \cap \mathcal{J}^B = \emptyset$. Each job J_j^X, $j = 1, 2, \ldots, n_X$, is identified by a processing time p_j^X and a weight w_j^X. All jobs arrive at time zero and must be executed without preemption on a single machine. Moreover, a set of n_X due dates $D^X = \{d_1^X, d_2^X, \ldots, d_{n_X}^X\}$ needs to be assigned to the n_X jobs of agent X. To facilitate presentation, we assume that the X-jobs are numbered in the *shortest processing time* (SPT) order (or sequence), i.e., $p_1^X \le p_2^X \le \cdots \le p_{n_X}^X$, and the due dates in D^X are numbered in the *earliest due date* (EDD) order (or sequence), i.e., $d_1^X \le d_2^X \le \cdots \le d_{n_X}^X$. All parameters p_j^X, w_j^X and d_j^X are assumed to be nonnegative integers.

A solution ϕ for the jobs in $\mathcal{J}^A \cup \mathcal{J}^B$ is defined by a due date assignment (i.e., assigning a due date in D^X to a specific X-job in $\mathcal{J}^X$) and a feasible schedule (i.e., determining the job processing sequence and the starting time of each job). In addition, for any given solution ϕ, the following notations are employed:

- $\mathcal{J} = \mathcal{J}^A \cup \mathcal{J}^B$: the set of all A-jobs and B-jobs;
- $n = n_A + n_B$: the number of jobs in $\mathcal{J}$;
- $P = \sum_{J_j^X \in \mathcal{J}} p_j^X$: the total processing time of jobs in $\mathcal{J}$;
- $d_{\max} = \max\{d_j^X : J_j^X \in \mathcal{J}\}$: the largest due date among all jobs in $\mathcal{J}$;
- $p_{\max} = \max\{p_j^X : J_j^X \in \mathcal{J}\}$: the largest processing time among all jobs in $\mathcal{J}$;
- $\mathcal{J}_j^X = \{J_1^X, J_2^X, \ldots, J_j^X\}$: the set of the first j X-jobs in $\mathcal{J}^X$;
- $D_j^X = \{d_1^X, d_2^X, \ldots, d_j^X\}$: the set of the first j due dates in D^X;
- $P_j^X = \sum_{i=1}^{j} p_i^X$: the total processing time of the jobs in $\mathcal{J}_j^X$;
- $D_j^X(\phi)$: the due date assigned to J_j^X in ϕ;
- $C_j^X(\phi)$: the completion time of J_j^X in ϕ;
- $T_j^X(\phi) = \max\{C_j^X(\phi) - D_j^X(\phi), 0\}$: the tardiness of J_j^X in ϕ;
- $U_j^X(\phi)$: the tardy indicator of J_j^X in ϕ, where $U_j^X(\phi) = 0$ if $C_j^X(\phi) \le D_j^X(\phi)$, and $U_j^X(\phi) = 1$ otherwise;
- $Y_j^X(\phi) = \min\{p_j^X, \max\{C_j^X(\phi) - D_j^X(\phi), 0\}\}$: the late work of J_j^X in ϕ;
- $f_j^X(\phi)$: the scheduling cost of J_j^X in ϕ;
- $f_{\max}^X = \max\{f_j^X(\phi) : 1 \le j \le n_X\}$: the maximum cost of agent X;
- $f_{\text{sum}}^X = \sum_{j=1}^{n_X} f_j^X(\phi)$: the total cost of agent X;
- $\sum T_j^X = \sum_{j=1}^{n_X} T_j^X(\phi)$: the total tardiness of agent X;
- $\sum Y_j^X = \sum_{j=1}^{n_X} Y_j^X(\phi)$: the total late work of agent X;
- $\sum w_j^X Y_j^X = \sum_{j=1}^{n_X} w_j^X Y_j^X(\phi)$: the total weighted late work of agent X;
- $\sum U_j^X = \sum_{j=1}^{n_X} U_j^X(\phi)$: the number of tardy jobs of agent X;
- $\sum w_j^X U_j^X = \sum_{j=1}^{n_X} w_j^X U_j^X(\phi)$: the weighted number of tardy jobs of agent X.

Let $f^X \in \{f^X_{\max}, f^X_{\text{sum}}\}$ be an arbitrary scheduling cost of agent X. f^X is called *regular* if it is nondecreasing in relation to all completion times of agent X. If $U^X_j(\phi) = 1$, then J^X_j is called *tardy*, otherwise it is called *non-tardy*. If $Y^X_j(\phi) = p^X_j$, then J^X_j is called *fully late*, otherwise it is called *non-fully late*. Note that $Y^X_j(\phi) \le T^X_j(\phi)$ and $Y^X_j(\phi) = T^X_j(\phi)$ if $C^X_j(\phi) \le D^X_j(\phi) + p^X_j$. When there is no confusion, the notations D^X_j, C^X_j, T^X_j, Y^X_j and U^X_j are used to denote $D^X_j(\phi)$, $C^X_j(\phi)$, $T^X_j(\phi)$, $Y^X_j(\phi)$ and $U^X_j(\phi)$, respectively.

The goal of this paper is to find a solution so that the weighted sum (i.e., linear combination) of the scheduling criteria f^A and f^B is minimized. Adopting the extended scheduling scheme proposed in Agnetis et al. (2014), the problems under study are denoted by $1|\text{ADD}|f^A + \lambda\sum Y^B_j$, where $f^A \in \{\sum Y^A_j, \sum w^A_j U^A_j, \sum T^A_j\}$ and $\lambda > 0$ is a fixed weighting factor. The linear combination method is chosen for balancing the objectives of two agents for two key reasons. First, it offers a simple and intuitive way to integrate both agents' objectives into a single objective function, which allows us to adjust the relative importance of these objectives by varying the parameter λ. Second, it provides a clear and interpretable trade-off between the two objectives, which can facilitate both theoretical analysis and practical implementation. By setting different values of λ, we can explore the trade-off between the objectives of agents A and B. Since all jobs arrive at time zero and all studied scheduling criteria are regular, we only need to cope with the solutions where each job begins its processing as early as possible. Hence, a feasible schedule can be totally defined by a permutation of jobs in $\mathcal{J}$. To simplify the discussion, in a given solution ϕ to $1|\text{ADD}|f^A + \lambda f^B$, J^X_j ($X \in \{A, B\}$) is called *effective* if (i) it is non-tardy (i.e., $C^X_j(\phi) \le D^X_j(\phi)$) when the criterion f^X is $\sum w^X_j U^X_j$, or (ii) it is non-fully late (i.e., $C^X_j(\phi) < D^X_j(\phi) + p^X_j$) when the criterion f^X is $\sum w^X_j Y^X_j$, or (iii) the criterion f^X is $\sum T^X_j$.

Because the single-agent problem $1|\text{ADD}|\sum Y^B_j$ is NP-hard (Mosheiov et al., 2021), all the general models studied in this paper are also NP-hard. For this reason, our focus is on establishing their computational complexity, i.e., on proving the binary NP-hardness. This is accomplished by constructing pseudo-polynomial-time algorithms for $1|\text{ADD}|f^A + \lambda\sum Y^B_j$ with $f^A \in \{\sum Y^A_j, \sum w^A_j U^A_j, \sum T^A_j\}$. Furthermore, we identify several polynomially solvable special cases. Specifically, we concentrate on scenarios where the processing times for particular agents are identical and where the job processing times and weights are antithetical, meaning that a job with a larger weight has a correspondingly smaller processing time compared to another job with a smaller weight.

3. Problem $1|\text{ADD}|f^A + \lambda\sum Y^B_j$

In this section, we start with establishing several fundamental and crucial properties of optimal solutions for the general problem $1|\text{ADD}|f^A+\lambda f^B$, where both f^A and f^B are regular scheduling criteria. Subsequently, we design pseudo-polynomial-time DP algorithms for $1|\text{ADD}|\sum Y^A_j + \lambda\sum Y^B_j$, $1|\text{ADD}|\sum w_j U^A_j + \lambda\sum Y^B_j$ and $1|\text{ADD}|\sum T^A_j + \lambda\sum Y^B_j$. Finally, we conduct extensive numerical experiments to test the performance of these DP algorithms.

For a given job subset $\mathcal{J}' \subseteq \mathcal{J}^X$ and a given due date subset $\mathcal{D}' \subseteq \mathcal{D}^X$ with $|\mathcal{J}'| = |\mathcal{D}'|$, a partial solution corresponding to $\mathcal{J}'$ and $\mathcal{D}'$ is called an *SPT-EDD solution* if the jobs in $\mathcal{J}'$ are executed in the SPT sequence and the due dates in $\mathcal{D}'$ are assigned to these jobs adhering to the EDD priority.

Let $f^B \in \{f^B_{\max}, f^B_{\text{sum}}\}$ be an arbitrary regular scheduling criterion. We state the properties from the viewpoint of agent A, but the results can be symmetrically mapped onto the scenario of agent B. In the context of $1 \parallel \sum Y_j$, it is established that in an optimal schedule the non-fully late jobs are sequenced first, followed by the fully late jobs in arbitrary order (Potts and Van Wassenhove, 1992). The following lemma demonstrates that an analogous result holds for $1|\text{ADD}|\sum w^A_j Y^A_j + \lambda f^B$ and the smallest due dates in D^A can be assigned to these fully late jobs.

Lemma 3.1. *There exists an optimal solution to* $1|\text{ADD}|\sum w^A_j Y^A_j + \lambda f^B$ *in which:*

(1) the fully late A-jobs are executed after all effective jobs in arbitrary order;

(2) if u A-jobs are fully late, then the due dates in D^A_u *can be assigned to these jobs arbitrarily.*

Proof. Let ϕ be an optimal solution to $1|\text{ADD}|\sum Y^A_j + \lambda f^B$ where Property (1) is violated. We construct another solution ϕ' from ϕ by shifting all fully late A-jobs after all effective jobs. Clearly, the completion times of all effective jobs do not increase in ϕ', we have $\sum w^A_j Y^A_j(\phi') + \lambda f^B(\phi') \le \sum w^A_j Y^A_j(\phi) + \lambda f^B(\phi)$. Hence, ϕ' is an optimal solution that satisfies Property (1).

Assume that ϕ is an optimal solution with u fully late A-jobs that satisfies Property (1) but does not satisfy Property (2). Then there exist an index $g \in \{1, 2, \dots, u\}$ and an index $h \in \{u+1, u+2, \dots, n_A\}$ such that d^A_g is assigned to an effective job J^A_x and d^A_h is assigned to a fully-late job J^A_y. We construct another solution ϕ' from ϕ by interchanging the due dates of jobs J^A_x and J^A_y. Note that the completion times of all jobs will not change in ϕ' and the assigned due date of J^A_x will not become smaller. Then we have $Y^A_x(\phi') \le Y^A_x(\phi)$ and $Y^A_y(\phi') = Y^A_y(\phi) = p^A_y$. Thus, we have $\sum w^A_j Y^A_j(\phi') + \lambda f^B(\phi') \le \sum w^A_j Y^A_j(\phi) + \lambda f^B(\phi)$. By repeating the above process successively, we can obtain an optimal solution that satisfies both Properties (1) and (2). □

The following lemma outlines the processing sequence and due date assignment strategy of the effective A-jobs in an optimal solution to $1|\text{ADD}|\sum Y^A_j + \lambda f^B$.

Lemma 3.2. *There exists an optimal solution to* $1|\text{ADD}|\sum Y^A_j + \lambda f^B$ *in which the effective A-jobs are executed in line with the SPT-EDD principle.*

Proof. Assume that ϕ is an optimal solution that satisfies the properties established in Lemma 3.1 but the effective A-jobs are not executed in line with the SPT-EDD principle. For simplicity, let ϕ be the optimal solution which is closest to the desired structure, i.e., the number of A-job pairs in wrong order and the number of due date pairs in wrong order is the least. Then, if the total number of such wrong pairs is positive, we differentiate between the following two cases.

Case 1: the effective A-jobs in ϕ are not executed according to the SPT sequence. Let ϕ be an optimal solution of the form $\phi = (D, J^A_x, E, J^A_y, F)$, where J^A_x and J^A_y are the first pair of effective A-jobs with $p^A_x > p^A_y$, and D, E and F are arbitrary (partial) sequences of jobs in $\mathcal{J}$. This implies that $C^A_x(\phi) < C^A_y(\phi)$, $Y^A_x(\phi) = \min\{p^A_x, T^A_x(\phi)\} = T^A_x(\phi) < p^A_x$ and $Y^A_y(\phi) = \min\{p^A_y, T^A_y(\phi)\} = T^A_y(\phi) < p^A_y$. We construct another solution ϕ' from ϕ by interchanging the positions of J^A_x and J^A_y and their due date assignments as well. Since $p^A_x > p^A_y$, we have $D^A_x(\phi') = D^A_y(\phi)$, $D^A_y(\phi') = D^A_x(\phi)$, $C^A_x(\phi') = C^A_y(\phi)$, $C^A_y(\phi') = C^A_x(\phi) - p^A_x + p^A_y$, $C^X_j(\phi') = C^X_j(\phi)$ for each $J^X_j \in D \cup F$, and $C^X_j(\phi') < C^X_j(\phi)$ for each $J^X_j \in E$. In this case, $T^A_x(\phi') = \max\{C^A_x(\phi') - D^A_x(\phi'), 0\} = \max\{C^A_y(\phi) - D^A_y(\phi), 0\} = T^A_y(\phi)$ and $T^A_y(\phi') = \max\{C^A_y(\phi') - D^A_y(\phi'), 0\} = \max\{C^A_x(\phi) - p^A_x + p^A_y - D^A_x(\phi), 0\} \le \max\{C^A_x(\phi) - D^A_x(\phi), 0\} = T^A_x(\phi)$. Then we have $Y^A_x(\phi') = \min\{p^A_x, T^A_x(\phi')\} = \min\{p^A_x, T^A_y(\phi)\} = \min\{p^A_x, Y^A_y(\phi)\} \le Y^A_y(\phi)$ and $Y^A_y(\phi') = \min\{p^A_y, T^A_y(\phi')\} \le \min\{p^A_y, T^A_x(\phi)\} = \min\{p^A_y, Y^A_x(\phi)\} \le Y^A_x(\phi)$. Due to the regularity of the considered scheduling criteria, we have $\sum Y^A_j(\phi') + \lambda f^B(\phi') \le \sum Y^A_j(\phi) + \lambda f^B(\phi)$.

Case 2: the effective A-jobs in ϕ are executed according to the SPT sequence but the due dates are not assigned to these jobs adhering to the EDD rule. As in Case 1, let ϕ be an optimal solution of the form $\phi = (D, J^A_x, E, J^A_y, F)$, where J^A_x and J^A_y are the first pair of effective A-jobs with $D^A_x(\phi) > D^A_y(\phi)$. This implies that $C^A_x(\phi) \le C^A_y(\phi) - p^A_y < D^A_y(\phi)$. Since $D^A_x(\phi) > D^A_y(\phi)$, we have $Y^A_x(\phi) = \min\{p^A_x, T^A_x(\phi)\} = T^A_x(\phi) = 0$ and $Y^A_y(\phi) = \min\{p^A_y, T^A_y(\phi)\} = T^A_y(\phi) < p^A_y$. We construct another solution ϕ' from ϕ by interchanging the due dates of jobs J^A_x and

J_y^A. In this case, $T_x^A(\phi') = \max\{C_x^A(\phi') - D_x^A(\phi'), 0\} = \max\{C_x^A(\phi) - D_x^A(\phi), 0\} = 0$ and $T_y^A(\phi') = \max\{C_y^A(\phi') - D_y^A(\phi'), 0\} = \max\{C_y^A(\phi) - D_y^A(\phi), 0\} \le \max\{C_y^A(\phi) - D_y^A(\phi), 0\} = T_y^A(\phi)$. Then we have $Y_x^A(\phi') = \min\{p_x^A, T_x^A(\phi')\} = 0$ and $Y_y^A(\phi') = \min\{p_y^A, T_y^A(\phi')\} \le \min\{p_y^A, T_y^A(\phi)\} = Y_y^A(\phi)$. Thus, we have $\sum Y_j^A(\phi') + \lambda f^B(\phi') \le \sum Y_j^A(\phi) + \lambda f^B(\phi)$.

In both cases, the proposed transformations show that we can decrease the number of wrong pairs while maintaining the solution optimality, which is a contradiction. □

When $f^B \in \{L_{\max}^B, \sum w_j^B U_j^B, \sum T_j^B\}$, the following two results, which are very crucial for our subsequent discussions, have been proved by Yin et al. (2012). In fact, their proofs hold for an arbitrary regular scheduling criterion as well.

Lemma 3.3. *There exists an optimal solution to* $1|\text{ADD}|\sum T_j^A + \lambda f^B$ *in which the A-jobs are executed in line with the SPT-EDD principle.*

Lemma 3.4. *There exists an optimal solution to* $1|\text{ADD}|\sum w_j^A U_j^A + \lambda f^B$ *in which:*

(1) the tardy A-jobs are executed after all effective jobs in arbitrary order;
(2) if u A-jobs are tardy, then the due dates in $\mathcal{D}_u^A$ *can be assigned to these jobs arbitrarily;*
(3) the effective A-jobs are executed in line with the SPT-EDD principle.

3.1. Problem $1|\text{ADD}|\sum Y_j^A + \lambda\sum Y_j^B$

In view of Lemmas 3.1 and 3.2, we direct our attention to the solutions (called *normal*) of $1|\text{ADD}|\sum Y_j^A + \lambda\sum Y_j^B$ that adhere to the following conditions: (i) the effective A-jobs and B-jobs are executed in line with their respective SPT-EDD principles; (ii) all fully late jobs are executed after all effective jobs in any arbitrary order; and (iii) if there are u_X fully late X-jobs, then the due dates in $\mathcal{D}_{u_X}^X$ can be assigned to these jobs arbitrarily, where $X \in \{A, B\}$. Let the vector (j_A, j_B, s, k_A, k_B) denote a state standing for the scenario where the set of jobs under consideration is $\mathcal{J} \setminus (\mathcal{J}_{j_A-1}^A \cup \mathcal{J}_{j_B-1}^B)$, the earliest starting time of these jobs is s, $0 \le s \le P_{j_A-1}^A + P_{j_B-1}^B$, k_A A-jobs are fully late, $0 \le k_A \le n_A - j_A + 1$, and k_B B-jobs are fully late, $0 \le k_B \le n_B - j_B + 1$. Define $G(j_A, j_B, s, k_A, k_B)$ to be the optimal solution value for state (j_A, j_B, s, k_A, k_B). If there is no such solution, then we define $G(j_A, j_B, s, k_A, k_B) = +\infty$.

With respect to a given state (j_A, j_B, s, k_A, k_B), let ϕ be an optimal normal solution that attains the value $G(j_A, j_B, s, k_A, k_B)$. Four situations associated with jobs $J_{j_A}^A$ and $J_{j_B}^B$ are analyzed.

Situation 1: $J_{j_A}^A$ is the first scheduled effective job in ϕ. Then the due date $d_{j_A+k_A}^A$ is assigned to $J_{j_A}^A$, i.e., $D_{j_A}^A(\phi) = d_{j_A+k_A}^A$. Thus, we have $Y_{j_A}^A(\phi) = \min\{p_{j_A}^A, \max\{0, s + p_{j_A}^A - d_{j_A+k_A}^A\}\} = \max\{0, s + p_{j_A}^A - d_{j_A+k_A}^A\}$. In this situation, the jobs that follow $J_{j_A}^A$ are linked to $(j_A + 1, j_B, s + p_{j_A}^A, k_A, k_B)$, and so we have $G(j_A, j_B, s, k_A, k_B) = G(j_A + 1, j_B, s + p_{j_A}^A, k_A, k_B) + \max\{0, s + p_{j_A}^A - d_{j_A+k_A}^A\}$. Note that this situation happens only when $s \le d_{j_A+k_A}^A - 1$.

Situation 2: $J_{j_A}^A$ is fully late in ϕ. Then we have $Y_{j_A}^A(\phi) = p_{j_A}^A$. In this situation, the jobs that follow $J_{j_A}^A$ are linked to $(j_A + 1, j_B, s, k_A - 1, k_B)$, and so we have $G(j_A, j_B, s, k_A, k_B) = G(j_A + 1, j_B, s, k_A - 1, k_B) + p_{j_A}^A$.

Situation 3: $J_{j_B}^B$ is the first scheduled effective job in ϕ. Similar to the analysis of Situation 1, it follows that $d_{j_B+k_B}^B$ is assigned to $J_{j_B}^B$ and $G(j_A, j_B, s, k_A, k_B) = G(j_A, j_B + 1, s + p_{j_B}^B, k_A, k_B) + \lambda\max\{0, s + p_{j_B}^B - d_{j_B+k_B}^B\}$. Note that this situation happens only when $s \le d_{j_B+k_B}^B - 1$.

Situation 4: $J_{j_B}^B$ is fully late in ϕ. Similar to the analysis of Situation 2, it follows that $G(j_A, j_B, s, k_A, k_B) = G(j_A, j_B + 1, s, k_A, k_B - 1) + \lambda p_{j_B}^B$.

Taking into account the preceding discussion, a DP algorithm (denoted $\mathbf{DP_1}$) is introduced for $1|\text{ADD}|\sum Y_j^A + \lambda\sum Y_j^B$.

The initial conditions are

$$G(n_A, n_B + 1, s, k_A, k_B)$$
$$= \begin{cases} p_{n_A}^A, & \text{if } (0 \le s \le P_{n_A-1}^A + P_{n_B}^B) \\ & \quad \wedge(k_A = 1) \wedge (k_B = 0); \\ \max\{0, s + p_{n_A}^A - d_{n_A}^A\}, & \text{if } (0 \le s \le \min\{P_{n_A-1}^A + P_{n_B}^B, d_{n_A}^A - 1\}) \\ & \quad \wedge(k_A = 0) \wedge (k_B = 0); \\ +\infty, & \text{otherwise;} \end{cases}$$

and

$$G(n_A + 1, n_B, s, k_A, k_B)$$
$$= \begin{cases} \lambda p_{n_B}^B, & \text{if } (0 \le s \le P_{n_A}^A + P_{n_B-1}^B) \\ & \quad \wedge(k_A = 0) \wedge (k_B = 1); \\ \lambda\max\{0, s + p_{n_B}^B - d_{n_B}^B\}, & \text{if } (0 \le s \le \min\{P_{n_A}^A + P_{n_B-1}^B, d_{n_B}^B - 1\}) \\ & \quad \wedge(k_A = 0) \wedge (k_B = 0); \\ +\infty, & \text{otherwise.} \end{cases}$$

The boundary conditions for $s = 0, 1, \ldots, P_{j_A-1}^A + P_{j_B-1}^B$ are

$$G(j_A, j_B, s, k_A, k_B) = +\infty, \text{ if } (\min\{k_A, k_B\} < 0) \vee (k_A > n_A - j_A + 1) \vee (k_B > n_B - j_B + 1).$$

The recursive relations for $j_A = n_A+1, n_A, \ldots, 1$, $j_B = n_B+1, n_B, \ldots, 1$, $j_A + j_B \le n_A + n_B$, $s = 0, 1, \ldots, P_{j_A-1}^A + P_{j_B-1}^B$, $k_A = 0, 1, \ldots, n_A - j_A + 1$, $k_B = 0, 1, \ldots, n_B - j_B + 1$ can be formulated as follows:

$$G(j_A, j_B, s, k_A, k_B) = \min \begin{cases} G(j_A + 1, j_B, s + p_{j_A}^A, k_A, k_B) \\ \quad + \max\{0, s + p_{j_A}^A - d_{j_A+k_A}^A\}, & \text{if } s \le d_{j_A+k_A}^A - 1; \\ G(j_A + 1, j_B, s, k_A - 1, k_B) + p_{j_A}^A; \\ G(j_A, j_B + 1, s + p_{j_B}^B, k_A, k_B) \\ \quad + \lambda\max\{0, s + p_{j_B}^B - d_{j_B+k_B}^B\}, & \text{if } s \le d_{j_B+k_B}^B - 1; \\ G(j_A, j_B + 1, s, k_A, k_B - 1) + \lambda p_{j_B}^B. \end{cases}$$

The optimum solution value is determined by

$$G^* = \min\{G(1, 1, 0, k_A, k_B) : 0 \le k_A \le n_A, 0 \le k_B \le n_B\}.$$

Theorem 3.1. *Problem* $1|\text{ADD}|\sum Y_j^A + \lambda\sum Y_j^B$ *can be solved by Algorithm* $\mathbf{DP_1}$ *in* $O(n_A^2 n_B^2 P)$ *time.*

Proof. The correctness of Algorithm $\mathbf{DP_1}$ stems from the properties established in Lemmas 3.1 and 3.2 and the above discussion. Note that $0 \le j_X \le n_X + 1$, $0 \le s \le P_{j_A-1}^A + P_{j_B-1}^B \le P$ and $0 \le k_X \le n_X - j_X + 1$ for $X = A, B$. Hence, at most $O(n_A^2 n_B^2 P)$ different states (j_A, j_B, s, k_A, k_B) need to be considered. Since each value $G(j_A, j_B, s, k_A, k_B)$ can be computed in constant time, the time complexity of $\mathbf{DP_1}$ is $O(n_A^2 n_B^2 P)$. □

Based on the result of Theorem 3.1 and the NP-hardness of $1|\text{ADD}|\sum Y_j$ (Mosheiov et al., 2021), it follows that problem $1|\text{ADD}|\sum Y_j^A + \lambda\sum Y_j^B$ is only binary NP-hard.

Corollary 3.1. *Problem* $1|\text{ADD}, p_j^A = p_j^B = p|\sum Y_j^A + \lambda\sum Y_j^B$ *can be solved in* $O(nn_A^2 n_B^2)$ *time.*

Proof. When all jobs have equal processing times, i.e., $p_j^A = p_j^B = p$, the starting time variable s in $\mathbf{DP_1}$ has only n possible values. Utilizing the proof of Theorem 3.1, the result holds. □

Note that by ignoring the variables defined for one out of two agents and considering the remaining ones for a single agent only, $\mathbf{DP_1}$ can be directly utilized to solve the counterpart single-agent problem. Hence, the following result holds.

Corollary 3.2. *Problem* $1|\text{ADD}|\sum Y_j$ *can be solved in* $O(n^2 P)$ *time.*

Remark 3.1. As the time complexities of the DP algorithms provided by Chen et al. (2023) and Justkowiak et al. (2023) for $1|\text{ADD}|\sum Y_j$ are both $O(n^3 P)$, the time complexity of $\mathbf{DP_1}$ is lower by the factor of n.

Example 3.1. Consider a 4-job instance of 1|ADD|$\sum Y_j^A + \lambda \sum Y_j^B$, in which $\mathcal{J}^A = \{J_1^A, J_2^A\}$, $\mathcal{J}^B = \{J_1^B, J_2^B\}$, $\mathcal{D}^A = \{3, 10\}$, $\mathcal{D}^B = \{4, 7\}$, the processing times of A-jobs are $p_1^A = 2$ and $p_2^A = 4$, the processing times of B-jobs are $p_1^B = 3$ and $p_2^B = 5$, and the weighting factor is $\lambda = 1$.

By applying Algorithm **DP$_1$**, we get an optimal solution ϕ in which the job processing sequence is given by $J_1^A \to J_2^B \to J_2^A \to J_1^B$, and the due date assignment strategy is given by $D_1^A(\phi) = 3$, $D_2^A(\phi) = 10$, $D_1^B(\phi) = 4$ and $D_2^B(\phi) = 7$. Then we have $Y_1^A(\phi) = 0$, $Y_2^A(\phi) = 1$, $Y_1^B(\phi) = 3$ and $Y_2^B(\phi) = 0$. This implies that J_1^A, J_2^A and J_2^B are effective jobs, and J_1^B is a fully late job. Hence, the optimal solution value is equal to $\sum_{j=1}^2 Y_j^A(\phi) + \lambda \sum_{j=1}^2 Y_j^B(\phi) = 1 + 3 = 4$. The detailed implementation of Example 3.1 consisting of all explicit state transitions of **DP$_1$** is illustrated in Appendix A.

3.2. Problem 1|ADD|$\sum w_j^A U_j^A + \lambda \sum Y_j^B$

In view of Lemmas 3.1, 3.2 and 3.4, we direct our attention to the solutions (called *normal*) of 1|ADD|$\sum w_j^A U_j^A + \lambda \sum Y_j^B$ that adhere to the following conditions: (i) the non-tardy A-jobs are executed in line with the SPT-EDD principle; (ii) the non-fully late B-jobs are executed in line with the SPT-EDD principle; (iii) all tardy A-jobs and fully late B-jobs are executed after all effective jobs; (iv) if u_A A-jobs are tardy, then the due dates in $\mathcal{D}_{u_A}^A$ can be assigned to these jobs arbitrarily; and (v) if u_B B-jobs are fully late, then the due dates in $\mathcal{D}_{u_B}^B$ can be assigned to these jobs arbitrarily. Let the vector (j_A, j_B, s, k_A, k_B) denote a state standing for the scenario where the set of jobs in consideration is $\mathcal{J} \setminus (\mathcal{J}_{j_A-1}^A \cup \mathcal{J}_{j_B-1}^B)$, the earliest starting time of these jobs is s, $0 \le s \le P_{j_A-1}^A + P_{j_B-1}^B$, k_A A-jobs are tardy, $0 \le k_A \le n_A - j_A + 1$, and k_B B-jobs are fully late, $0 \le k_B \le n_B - j_B + 1$. Define $G(j_A, j_B, s, k_A, k_B)$ to be the optimal solution value for state (j_A, j_B, s, k_A, k_B). If there is no such solution, then we define $G(j_A, j_B, s, k_A, k_B) = +\infty$.

With respect to a given state (j_A, j_B, s, k_A, k_B), let ϕ be an optimal normal solution that attains the value $G(j_A, j_B, s, k_A, k_B)$. Four situations associated with jobs $J_{j_A}^A$ and $J_{j_B}^B$ are analyzed.

Situation 1: $J_{j_A}^A$ is the first scheduled effective job in ϕ. Then the due date $d_{j_A+k_A}^A$ is assigned to $J_{j_A}^A$, i.e., $D_{j_A}^A(\phi) = d_{j_A+k_A}^A$. In this situation, the jobs that follow $J_{j_A}^A$ are linked to $(j_A+1, j_B, s+p_{j_A}^A, k_A, k_B)$, and so we have $G(j_A, j_B, s, k_A, k_B) = G(j_A + 1, j_B, s + p_{j_A}^A, k_A, k_B)$. Note that this situation happens only when $s \le d_{j_A+k_A}^A - p_{j_A}^A$.

Situation 2: $J_{j_A}^A$ is tardy in ϕ. In this situation, the jobs that follow $J_{j_A}^A$ are linked to $(j_A + 1, j_B, s, k_A - 1, k_B)$, and so we have $G(j_A, j_B, s, k_A, k_B) = G(j_A + 1, j_B, s, k_A - 1, k_B) + w_{j_A}^A$.

Situation 3: $J_{j_B}^B$ is the first scheduled effective job in ϕ. Then the due date $d_{j_B+k_B}^B$ is assigned to $J_{j_B}^B$, i.e., $D_{j_B}^B(\phi) = d_{j_B+k_B}^B$. Thus, we have $Y_{j_B}^B(\phi) = \max\{0, s + p_{j_B}^B - d_{j_B+k_B}^B\}$. In this situation, the jobs that follow $J_{j_B}^B$ are linked to $(j_A, j_B + 1, s + p_{j_B}^B, k_A, k_B)$, and so we have $G(j_A, j_B, s, k_A, k_B) = G(j_A, j_B+1, s+p_{j_B}^B, k_A, k_B)+\lambda \max\{0, s+p_{j_B}^B-d_{j_B+k_B}^B\}$. Note that this situation happens only when $s \le d_{j_B+k_B}^B - 1$

Situation 4: $J_{j_B}^B$ is fully late in ϕ. Then we have $Y_{j_B}^B(\phi) = p_{j_B}^B$. In this situation, the jobs that follow $J_{j_B}^B$ are linked to $(j_A, j_B + 1, s, k_A, k_B - 1)$, and so we have $G(j_A, j_B, s, k_A, k_B) = G(j_A, j_B + 1, s, k_A, k_B - 1) + \lambda p_{j_B}^B$.

Taking into account the preceding discussion, a DP algorithm (denoted **DP$_2$**) is introduced for 1|ADD|$\sum w_j^A U_j^A + \lambda \sum Y_j^B$.

The initial conditions are

$$G(n_A, n_B + 1, s, k_A, k_B) = \begin{cases} w_{n_A}^A, & \text{if } (0 \le s \le P_{n_A-1}^A + P_{n_B}^B) \wedge (k_A = 1) \wedge (k_B = 0); \\ 0, & \text{if } (0 \le s \le \min\{P_{n_A-1}^A + P_{n_B}^B, d_{n_A}^A - p_{n_A}^A\}) \wedge (k_A = 0) \wedge (k_B = 0); \\ +\infty, & \text{otherwise;} \end{cases}$$

and

$$G(n_A + 1, n_B, s, k_A, k_B) = \begin{cases} \lambda p_{n_B}^B, & \text{if } (0 \le s \le P_{n_A}^A + P_{n_B-1}^B) \\ & \quad \wedge (k_A = 0) \wedge (k_B = 1); \\ \lambda \max\{0, s + p_{n_B}^B - d_{n_B}^B\}, & \text{if } (0 \le s \le \min\{P_{n_A}^A + P_{n_B-1}^B, d_{n_B}^B - 1\}) \\ & \quad \wedge (k_A = 0) \wedge (k_B = 0); \\ +\infty, & \text{otherwise.} \end{cases}$$

The boundary conditions for $s = 0, 1, \ldots, P_{j_A-1}^A + P_{j_B-1}^B$ are

$$G(j_A, j_B, s, k_A, k_B) = +\infty, \text{ if } (\min\{k_A, k_B\} < 0) \vee (k_A > n_A - j_A + 1) \vee (k_B > n_B - j_B + 1).$$

The recursive relations for $j_A = n_A+1, n_A, \ldots, 1$, $j_B = n_B+1, n_B, \ldots, 1$, $j_A + j_B \le n_A + n_B$, $s = 0, 1, \ldots, P_{j_A-1}^A + P_{j_B-1}^B$, $k_A = 0, 1, \ldots, n_A - j_A + 1$, $k_B = 0, 1, \ldots, n_B - j_B + 1$ can be formulated as follows:

$$G(j_A, j_B, s, k_A, k_B) = \min \begin{cases} G(j_A + 1, j_B, s + p_{j_A}^A, k_A, k_B), & \text{if } s \le d_{j_A+k_A}^A - p_{j_A}^A; \\ G(j_A + 1, j_B, s, k_A - 1, k_B) + w_{j_A}^A; & \\ G(j_A, j_B + 1, s + p_{j_B}^B, k_A, k_B) & \\ \quad + \lambda \max\{0, s + p_{j_B}^B - d_{j_B+k_B}^B\}, & \text{if } s \le d_{j_B+k_B}^B - 1; \\ G(j_A, j_B + 1, s, k_A, k_B - 1) + \lambda p_{j_B}^B. & \end{cases}$$

The optimum solution value is determined by

$$G^* = \min\{G(1, 1, 0, k_A, k_B) : 0 \le k_A \le n_A, 0 \le k_B \le n_B\}.$$

Theorem 3.2. *Problem* 1|ADD|$\sum w_j^A U_j^A + \lambda \sum Y_j^B$ *can be solved by Algorithm* **DP$_2$** *in* $O(n_A^2 n_B^2 P)$ *time.*

Proof. The correctness of Algorithm **DP$_2$** stems from the properties established in Lemmas 3.1, 3.2 and 3.4 and the above discussion. Note that $0 \le j_X \le n_X + 1$, $0 \le s \le P_{j_A-1}^A + P_{j_B-1}^B \le P$ and $0 \le k_X \le n_X - j_X + 1$ for $X = A, B$. Hence, at most $O(n_A^2 n_B^2 P)$ different states (j_A, j_B, s, k_A, k_B) need to be considered. Since each value $G(j_A, j_B, s, k_A, k_B)$ can be computed in constant time, the time complexity of **DP$_2$** is $O(n_A^2 n_B^2 P)$. □

Based on the result of Theorem 3.2 and the NP-hardness of 1|ADD|$\sum Y_j$ (Mosheiov et al., 2021) and 1|ADD|$\sum w_j U_j$ (Qi et al., 2002), it follows that problem 1|ADD|$\sum w_j^A U_j^A + \lambda \sum Y_j^B$ is only binary NP-hard.

Example 3.2. Consider a 6-job instance of 1|ADD|$\sum w_j^A U_j^A + \lambda \sum Y_j^B$, in which $\mathcal{J}^A = \{J_1^A, J_2^A, J_3^A\}$, $\mathcal{J}^B = \{J_1^B, J_2^B, J_3^B\}$, $\mathcal{D}^A = \{4, 7, 16\}$, $\mathcal{D}^B = \{5, 9, 20\}$, the processing times of A-jobs are $p_1^A = 2$, $p_2^A = 4$ and $p_3^A = 6$, the weights of A-jobs are $w_1^A = 5$, $w_2^A = 6$ and $w_3^A = 13$, the processing times of B-jobs are $p_1^B = 3$, $p_2^B = 5$ and $p_3^B = 7$, and the weighting factor is $\lambda = 2$.

By applying Algorithm **DP$_2$**, we get an optimal solution ϕ in which the job processing sequence is given by $J_1^A \to J_2^B \to J_3^A \to J_3^B \to J_2^A \to J_1^B$, and the due date assignment strategy is given by $D_1^A(\phi) = 7$, $D_2^A(\phi) = 4$, $D_3^A(\phi) = 16$, $D_1^B(\phi) = 5$, $D_2^B(\phi) = 9$ and $D_3^B(\phi) = 20$. Then we have $U_1^A(\phi) = 0$, $U_2^A(\phi) = 1$, $U_3^A(\phi) = 0$, $Y_1^B(\phi) = 3$, $Y_2^B(\phi) = 0$ and $Y_3^B(\phi) = 0$. This implies that J_1^A, J_3^A, J_2^B and J_3^B are effective jobs, J_2^A is a tardy job, and J_1^B is a fully late job. Hence, the optimal solution value is equal to $\sum_{j=1}^3 w_j^A U_j^A(\phi) + 2\sum_{j=1}^3 Y_j^B(\phi) = 6 + 6 = 12$. Since the calculations of **DP$_2$** are based on the framework similar to that of **DP$_1$**, we do not present its detailed calculations.

3.3. Problem 1|ADD|$\sum T_j^A + \lambda \sum Y_j^B$

In view of Lemmas 3.1–3.3, we direct our attention to solutions (called *normal*) of 1|ADD|$\sum T_j^A + \lambda \sum Y_j^B$ that adhere to the following conditions: (i) the A-jobs are executed in line with the SPT-EDD principle; (ii) the effective B-jobs are executed in line with the SPT-EDD principle; (iii) all fully late B-jobs are executed after all effective jobs;

and (iv) if there are u_B fully late B-jobs, then the due dates in $\mathcal{D}^B_{u_B}$ can be assigned to these jobs arbitrarily. Let the vector (j_A, j_B, s, k_B) denote a state standing for the scenario where the set of jobs in consideration is $\mathcal{J}\setminus(\mathcal{J}^A_{j_A-1}\cup\mathcal{J}^B_{j_B-1})$, the earliest starting time of these jobs is s, $P^A_{j_A-1}\le s\le P^A_{j_A-1}+P^B_{j_B-1}$, and k_B B-jobs are fully late, $0\le k_B\le n_B-j_B+1$. Define $G(j_A, j_B, s, k_B)$ to be the optimal solution value for state (j_A, j_B, s, k_B). If there is no such solution, then we define $G(j_A, j_B, s, k_B)=+\infty$.

With respect to a given state (j_A, j_B, s, k_B), let ϕ be an optimal normal solution that attains the value $G(j_A, j_B, s, k_B)$. Three situations associated with jobs $J^A_{j_A}$ and $J^B_{j_B}$ are analyzed.

Situation 1: $J^A_{j_A}$ is the first scheduled effective job in ϕ. Then we have $T^A_{j_A}(\phi)=\max\{0, s+p^A_{j_A}-d^A_{j_A}\}$. In this situation, the jobs that follow $J^A_{j_A}$ are linked to $(j_A+1, j_B, s+p^A_{j_A}, k_B)$, and so we have $G(j_A, j_B, s, k_B)=G(j_A+1, j_B, s+p^A_{j_A}, k_B)+\max\{0, s+p^A_{j_A}-d^A_{j_A}\}$.

Situation 2: $J^B_{j_B}$ is the first scheduled effective job in ϕ. Then the due date $d^B_{j_B+k_B}$ is assigned to $J^B_{j_B}$, i.e., $D^B_{j_B}(\phi)=d^B_{j_B+k_B}$. Thus, we have $Y^B_{j_B}(\phi)=\max\{0, s+p^B_{j_B}-d^B_{j_B+k_B}\}$. In this situation, the jobs that follow $J^B_{j_B}$ are linked to $(j_A, j_B+1, s+p^B_{j_B}, k_B)$, and so we have $G(j_A, j_B, s, k_B)=G(j_A, j_B+1, s+p^B_{j_B}, k_B)+\lambda\max\{0, s+p^B_{j_B}-d^B_{j_B+k_B}\}$. Note that this situation happens only when $s\le d^B_{j_B+k_B}-1$.

Situation 3: $J^B_{j_B}$ is fully late in ϕ. In this situation, the jobs that follow $J^B_{j_B}$ are linked to (j_A, j_B+1, s, k_B-1), and so we have $G(j_A, j_B, s, k_B)=G(j_A, j_B+1, s, k_B-1)+\lambda p^B_{j_B}$.

Taking into account the preceding discussion, a DP algorithm (denoted **DP$_3$**) is introduced for $1|\text{ADD}|\sum T^A_j+\lambda\sum Y^B_j$.

The initial conditions are

$$G(n_A, n_B+1, s, k_B)=\begin{cases}\max\{0, s+p^A_{n_A}-d^A_{n_A}\}, & \text{if } (P^A_{n_A-1}\le s\le P^A_{n_A-1}+P^B_{n_B})\wedge(k_B=0);\\ +\infty, & \text{otherwise;}\end{cases}$$

and

$$G(n_A+1, n_B, s, k_B)=\begin{cases}\lambda p^B_{n_B}, & \text{if } (\sum_{j=1}^{n_A}p^{n_A}_j\le s\le P^A_{n_A}+P^B_{n_B-1})\\ & \quad\wedge(k_B=1);\\ \lambda\max\{0, s+p^B_{n_B}-d^B_{n_B}\}, & \text{if } (0\le s\le\min\{P^A_{n_A}+P^B_{n_B-1}, d^B_{n_B}-1\})\\ & \quad\wedge(k_B=0);\\ +\infty, & \text{otherwise.}\end{cases}$$

The boundary conditions for $s=0,1,\ldots,P^A_{j_A-1}+P^B_{j_B-1}$ are

$$G(j_A, j_B, s, k_B)=+\infty,\ \text{if } (k_B<0)\vee(k_B>n_B-j_B+1).$$

The recursive relations for $j_A=n_A+1, n_A,\ldots,1$, $j_B=n_B+1, n_B,\ldots,1$, $j_A+j_B\le n_A+n_B$, $s=P^A_{j_A-1}, P^A_{j_A-1}+1,\ldots,P^A_{j_A-1}+P^B_{j_B-1}$, $k_B=0,1,\ldots,n_B-j_B+1$ can be formulated as follows:

$$G(j_A, j_B, s, k_B)=\min\begin{cases}G(j_A+1, j_B, s+p^A_{j_A}, k_B)+\max\{0, s+p^A_{j_A}-d^A_{j_A}\};\\ G(j_A, j_B+1, s+p^B_{j_B}, k_B)\\ \quad+\lambda\max\{0, s+p^B_{j_B}-d^B_{j_B+k_B}\}, & \text{if } s\le d^B_{j_B+k_B}-1;\\ G(j_A, j_B+1, s, k_B-1)+\lambda p^B_{j_B}.\end{cases}$$

The optimum solution value is determined by

$$G^*=\min\{G(1,1,0,k_B): 0\le k_B\le n_B\}.$$

Theorem 3.3. *Problem $1|\text{ADD}|\sum T^A_j+\lambda\sum Y^B_j$ can be solved by Algorithm* **DP$_3$** *in $O(n_An^2_BP)$ time.*

Proof. The correctness of Algorithm **DP$_3$** stems from the properties established in Lemmas 3.1–3.3 and the above discussion. Note that $0\le j_A\le n_A+1$, $0\le j_B\le n_B+1$, $0\le s\le P^A_{j_A-1}+P^B_{j_B-1}\le P$ and $0\le k_B\le n_B-j_B+1$. Hence, at most $O(n_An^2_BP)$ different states (j_A, j_B, s, k_B) need to be considered. Since each value $G(j_A, j_B, s, k_B)$ can be computed in constant time, the time complexity of **DP$_3$** is $O(n_An^2_BP)$. □

Based on the result of Theorem 3.3 and the NP-hardness of $1|\text{ADD}|\sum Y_j$ (Mosheiov et al., 2021), it follows that problem $1|\text{ADD}|\sum T^A_j+\lambda\sum Y^B_j$ is only binary NP-hard.

Example 3.3. Consider a 6-job instance of $1|\text{ADD}|\sum T^A_j+\lambda\sum Y^B_j$, in which $\mathcal{J}^A=\{J^A_1, J^A_2, J^A_3\}$, $\mathcal{J}^B=\{J^B_1, J^B_2, J^B_3\}$, $\mathcal{D}^A=\{3, 8, 17\}$, $\mathcal{D}^B=\{4, 10, 19\}$, the processing times of A-jobs are $p^A_1=2$, $p^A_2=4$ and $p^A_3=6$, the processing times of B-jobs are $p^B_1=3$, $p^B_2=5$ and $p^B_3=7$, and the weighting factor is $\lambda=2$.

By applying Algorithm **DP$_3$**, we get an optimal solution ϕ in which the job processing sequence is given by $J^A_1\to J^A_2\to J^B_2\to J^B_3\to J^A_3\to J^B_1$, and the due date assignment strategy is given by $D^A_1(\phi)=3$, $D^A_2(\phi)=8$, $D^A_3(\phi)=17$, $D^B_1(\phi)=4$, $D^B_2(\phi)=10$ and $D^B_3(\phi)=19$. Then we have $T^A_1(\phi)=0$, $T^A_2(\phi)=0$, $T^A_3(\phi)=7$, $Y^B_1(\phi)=3$, $Y^B_2(\phi)=1$ and $Y^B_3(\phi)=0$. This implies that J^B_2 and J^B_3 are effective jobs, J^B_1 is a fully late job. Hence, the optimal solution value is equal to $\sum_{j=1}^3 T^A_j(\phi)+2\sum_{j=1}^2 Y^B_j(\phi)=7+8=15$.

3.4. Numerical study

To test the performance of the DP algorithms developed in Sections 3.1-3.3, extensive numerical experiments are performed. The Matlab programs are run on a computer with an Intel(R) Core(TM) i7-8700 CPU @ 3.20 GHz with 8 GB RAM.

We separately evaluate the computational complexity of Algorithms **DP$_1$**-**DP$_3$** by measuring their running times across multiple test instances. These instances are generated using different combinations of input parameters. Specifically, following the data generation framework proposed by Hall and Posner (2001), the assumptions outlined below were used to generate a variety of parameter value groups.

(1) The number of jobs was $n\in\{20, 30, 40, 50, 60\}$ for **DP$_1$** and **DP$_2$**, while the number of jobs was $n\in\{20, 50, 80, 100, 120\}$ for **DP$_3$**, where $n_A=\lfloor\delta n\rfloor$, $n_B=n-n_A$, $\delta\in\{\frac{1}{3}, \frac{1}{2}, \frac{2}{3}\}$.

(2) The processing time p^X_j of job J^X_j was randomly and uniformly sampled from the set of integers in the interval $[1, p_{\max}]$, where $p_{\max}\in\{20, 50, 100\}$.

(3) The due date d^X_i was randomly and uniformly sampled from the set of integers in the interval $[1, \eta P]$, where $\eta\in\{0.4, 0.8\}$.

(4) The weight w^X_j of job J^X_j was randomly and uniformly sampled from the set of integers in the interval $[1, 50]$.

Consistent with the classifications in Atsmony et al. (2023) and Li and Liu (2024), instances with $n\le 30$ are categorized as small-sized instances, while those with $n=40, 50, 60, 80, 100$ are referred to as medium-sized instances. The selection of the parameter λ was determined based on the specific forms of the objective functions $f^A+\lambda\sum Y^B_j$, where $f^A\in\{\sum Y^A_j, \sum w^A_jU^A_j, \sum T^A_j\}$. While the choice of the weighting parameter λ does not affect the computational times, a reasonable setting of λ is important for preserving the two-agent nature of our model. For example, an extreme imbalance (e.g., $\lambda\to 0$) would essentially reduce the problem to optimizing agent A's objective alone, potentially rendering agent B's performance unacceptable. Conversely, excessively large values of λ would prioritize agent B' objective at the expense of agent A's critical operational requirements. To ensure that both agents' interests are meaningfully represented in the objective function, the values of λ are set to 1, 2, and 2 for **DP$_1$** (for $1|\text{ADD}|\sum Y^A_j+\lambda\sum Y^B_j$), **DP$_2$** (for $1|\text{ADD}|\sum w^A_jU^A_j+\lambda\sum Y^B_j$), and **DP$_3$** (for $1|\text{ADD}|\sum T^A_j+\lambda\sum Y^B_j$), respectively. These specific values are chosen based on the relative scales and sensitivities of the involved

Table 1
Average and maximum running times (sec) of $\mathbf{DP_1}$.

n	δ	η	$p_{\max}=20$		$p_{\max}=50$		$p_{\max}=100$	
			avg	max	avg	max	avg	max
20	$\frac{1}{3}$	0.4	0.0084	0.0112	0.0213	0.0285	0.0413	0.0492
20	$\frac{1}{3}$	0.8	0.0113	0.0143	0.0277	0.0382	0.0605	0.0890
20	$\frac{1}{2}$	0.4	0.0110	0.0136	0.0266	0.0484	0.0608	0.0860
20	$\frac{1}{2}$	0.8	0.0154	0.0232	0.0388	0.0537	0.0822	0.1046
20	$\frac{2}{3}$	0.4	0.0097	0.0169	0.0224	0.0311	0.0466	0.0703
20	$\frac{2}{3}$	0.8	0.0138	0.0195	0.0316	0.0382	0.0655	0.1066
30	$\frac{1}{3}$	0.4	0.0637	0.0836	0.1851	0.2346	0.3949	0.5099
30	$\frac{1}{3}$	0.8	0.0835	0.1164	0.2539	0.4548	0.5313	0.8376
30	$\frac{1}{2}$	0.4	0.0909	0.1299	0.2934	0.4069	0.6138	0.8129
30	$\frac{1}{2}$	0.8	0.1633	0.2807	0.4483	0.6097	0.9757	1.2552
30	$\frac{2}{3}$	0.4	0.0630	0.0821	0.1791	0.2613	0.3885	0.5011
30	$\frac{2}{3}$	0.8	0.0823	0.1061	0.2287	0.3512	0.5172	0.6733
40	$\frac{1}{3}$	0.4	0.3175	0.4294	0.8369	1.0725	1.9671	2.7495
40	$\frac{1}{3}$	0.8	0.4739	0.7546	1.1528	1.4917	2.6629	3.6271
40	$\frac{1}{2}$	0.4	0.4251	0.5867	1.1637	1.5607	2.5735	3.7517
40	$\frac{1}{2}$	0.8	0.5593	0.8597	1.4962	2.0943	2.9924	3.8458
40	$\frac{2}{3}$	0.4	0.3231	0.4359	0.9551	1.1469	1.9975	2.4390
40	$\frac{2}{3}$	0.8	0.4388	0.5649	1.1961	1.6354	2.5882	3.4903
50	$\frac{1}{3}$	0.4	1.0853	1.4500	2.9447	4.1387	6.5394	7.9854
50	$\frac{1}{3}$	0.8	1.3626	1.7286	3.8340	5.1142	10.3367	36.6892
50	$\frac{1}{2}$	0.4	1.3968	1.9070	3.5935	4.8457	8.4871	10.5286
50	$\frac{1}{2}$	0.8	1.8318	2.4657	4.5964	5.6002	124.0884	287.2276
50	$\frac{2}{3}$	0.4	1.1139	1.3881	2.9823	3.4898	6.7052	8.0238
50	$\frac{2}{3}$	0.8	1.4422	2.0641	3.8376	4.8508	14.1867	66.9022
60	$\frac{1}{3}$	0.4	1.7787	2.5262	5.9484	7.4455	–	–
60	$\frac{1}{3}$	0.8	2.2585	2.8370	16.4233	28.5968	–	–
60	$\frac{1}{2}$	0.4	2.2305	3.2999	9.0209	15.0424	–	–
60	$\frac{1}{2}$	0.8	3.0558	4.8824	–	–	–	–
60	$\frac{2}{3}$	0.4	1.6381	2.3652	5.3469	7.9855	–	–
60	$\frac{2}{3}$	0.8	2.1823	2.9711	14.040044	26.5968	–	–

objectives. In $\mathbf{DP_1}$, the objectives $\sum Y_j^A$ and $\sum Y_j^B$ are structurally similar and typically of comparable magnitude. This symmetry justifies our choice of $\lambda = 1$, which assigns equal weight to both agents' contributions. In $\mathbf{DP_2}$ and $\mathbf{DP_3}$, the weighted objective $\sum w_j^A U_j^A$ and total tardiness $\sum T_j^A$ tend to be larger in scale or more sensitive to changes compared to $\sum Y_j^B$. To address this imbalance and guarantee that agent B's objective maintains appropriate influence in the optimization process, we employ $\lambda = 2$ for these cases. For each group of n, $p_{\max}$, δ and η, we generate and solve 20 instances. Tables 1–3 report the average and maximum running times (in seconds) per instance set for $\mathbf{DP_1}$, $\mathbf{DP_2}$ and $\mathbf{DP_3}$, respectively. Note that for some problem instances, the memory usage exceeds the computer's maximum array size (i.e., the limit of 7.8 GB), making it impossible to solve them. Such experiments are marked with "–" in Tables 1–3.

The computational results in Tables 1 and 2 confirm that the running times are increasing functions of n, $p_{\max}$ and η. Despite the relatively high complexity $O(n_A^2 n_B^2 P)$ of $\mathbf{DP_1}$ and $\mathbf{DP_2}$, the results indicate that the actual running times are very small when the number of jobs n does not exceed 40 (in fact, the average running times do not exceed 5 s). For $n = 50$, the worst case occurs when $p_{\max} = 100$, $\delta = 0.5$, $\eta = 0.8$, the maximum running times for the generated 20 instances are about 287 s for $\mathbf{DP_1}$ and 383 s for $\mathbf{DP_2}$. However, the results imply that $\mathbf{DP_1}$ and $\mathbf{DP_2}$ cannot solve all 60-job instances due to the memory limit.

The computational results in Table 3 indicate that Algorithm $\mathbf{DP_3}$ is applicable for medium-size instances and the running times increase sharply as n and $p_{\max}$ increase. In fact, the actual running times of $\mathbf{DP_3}$ do not exceed 4 s for $n \le 80$, the maximum running time of 100-job instances does not exceed 17 s except for the worst case where $p_{\max} = 100$ and $\delta = \frac{1}{3}$, which requires about 161 s. Due to the memory limit, $\mathbf{DP_3}$ cannot solve all 120-job instances for $p_{\max} = 100$, but it can solve all 120-instances within 15 s on average for $p_{\max} = 50$.

4. Polynomially solvable cases

In this section, we identify various polynomially solvable cases of the models under consideration. One significant case is the scheduling problem where the processing times and weights of A-jobs exhibit an antithetical relationship, denoted $\mathrm{Ant}(p_j^A, w_j^A)$. This means that an A-job with a larger weight has a correspondingly smaller processing time compared to an A-job with a smaller weight. The concept of antithetical feature was first introduced in Dereniowski and Kubiak (2018) in the framework of shared machine scheduling. One motivation for this feature stems from the practical requirement to prioritize customer orders by assigning larger weights to those with smaller processing times. Second, from a theoretical viewpoint within the framework of two-agent scheduling, the antithetical feature encompasses special cases where the processing times of A-jobs are equal or the weights of A-jobs are equal (e.g., Oron et al., 2015; Fomin and Goldengorin, 2022; Chen et al., 2022a). Third, exploring various specific special cases can help refine the borderline between polynomial-time tractability and NP-hardness.

For ease of presentation, when the processing times and weights of A-jobs are antithetical, we assume that the jobs in $\mathcal{J}^A$ are indexed such that $p_1^A \le p_2^A \le \cdots \le p_{n_A}^A$ and $w_1^A \ge w_2^A \ge \cdots \ge w_{n_A}^A$ in this section. Moreover, for $X \in \{A, B\}$ and $j = 1, 2, \ldots, n_X$, let $J_{[j]}^X$ denote the jth completed X-job in any feasible schedule.

Lemma 4.1. *There exists an optimal solution to* $1|\mathrm{ADD}, \mathrm{Ant}(p_j^A, w_j^A)| \sum w_j^A U_j^A + \lambda f^B$ *in which the jobs in* $\mathcal{J}_{v_A}^A$ *are non-tardy for some* $v_A \in \{0, 1, \ldots, n_A\}$ *and the other A-jobs are tardy.*

Proof. Let ϕ be an optimal solution with v_A non-tardy A-jobs that satisfies the properties established in Lemma 3.4. Clearly, the result holds if $v_A = 0$ or $v_A = n_A$. Assume that $1 \le v_A \le n_A - 1$ and the v_A non-tardy A-jobs are not $J_1^A, J_2^A, \ldots, J_{v_A}^A$ in ϕ. Then there exists a pair

Table 2
Average and maximum running times (sec) of $\mathbf{DP_2}$.

n	δ	η	$p_{\max} = 20$		$p_{\max} = 50$		$p_{\max} = 100$	
			avg	max	avg	max	avg	max
20	$\frac{1}{3}$	0.4	0.0105	0.0149	0.0263	0.0397	0.0511	0.0780
20	$\frac{1}{3}$	0.8	0.0128	0.0209	0.0372	0.0465	0.0813	0.1302
20	$\frac{1}{2}$	0.4	0.0130	0.0171	0.0308	0.0452	0.0662	0.0993
20	$\frac{1}{2}$	0.8	0.0187	0.0271	0.0491	0.0849	0.1163	0.1651
20	$\frac{2}{3}$	0.4	0.0122	0.0187	0.0272	0.0388	0.0574	0.1105
20	$\frac{2}{3}$	0.8	0.0170	0.0255	0.0418	0.0714	0.0924	0.1356
30	$\frac{1}{3}$	0.4	0.0638	0.0816	0.2044	0.2624	0.4421	0.5731
30	$\frac{1}{3}$	0.8	0.1010	0.1733	0.3312	0.4242	0.6874	0.9048
30	$\frac{1}{2}$	0.4	0.1472	0.2037	0.3980	0.5553	0.8748	1.1143
30	$\frac{1}{2}$	0.8	0.2304	0.3018	0.5990	0.7531	1.3307	1.9281
30	$\frac{2}{3}$	0.4	0.0708	0.0948	0.2169	0.3065	0.4899	0.6730
30	$\frac{2}{3}$	0.8	0.1210	0.1824	0.3537	0.4458	0.7637	1.0353
40	$\frac{1}{3}$	0.4	0.3790	0.4398	0.9223	1.1069	2.1782	3.0962
40	$\frac{1}{3}$	0.8	0.4839	0.6031	1.1841	1.5359	2.6924	3.2977
40	$\frac{1}{2}$	0.4	0.5147	0.6265	1.2697	1.7149	3.2710	4.2257
40	$\frac{1}{2}$	0.8	0.6485	0.7813	1.6997	2.1811	3.7717	4.7531
40	$\frac{2}{3}$	0.4	0.4146	0.5465	1.0284	1.3007	2.5602	3.4161
40	$\frac{2}{3}$	0.8	0.5162	0.6890	1.3524	1.8246	3.1155	4.3933
50	$\frac{1}{3}$	0.4	1.1938	1.5379	3.3683	4.3859	7.5239	10.3981
50	$\frac{1}{3}$	0.8	1.4750	1.9578	4.2981	5.6591	20.1175	49.2866
50	$\frac{1}{2}$	0.4	1.6159	2.1571	4.5433	6.3005	9.7608	13.8228
50	$\frac{1}{2}$	0.8	1.9681	2.7191	6.8119	8.1527	190.4451	383.0845
50	$\frac{2}{3}$	0.4	1.3620	1.6437	3.6648	4.6713	7.8870	9.9753
50	$\frac{2}{3}$	0.8	2.1249	2.4447	5.3631	7.2388	16.2164	67.4168
60	$\frac{1}{3}$	0.4	3.5477	4.6762	10.4823	13.4760	–	–
60	$\frac{1}{3}$	0.8	4.3473	5.4312	22.7644	26.5994	–	–
60	$\frac{1}{2}$	0.4	4.9871	6.8359	15.5296	23.1047	–	–
60	$\frac{1}{2}$	0.8	5.9924	7.6413	–	–	–	–
60	$\frac{2}{3}$	0.4	3.4389	4.6187	11.0772	14.9382	–	–
60	$\frac{2}{3}$	0.8	5.1340	6.6276	23.6458	28.1319	–	–

Table 3
Average and maximum running times (sec) of $\mathbf{DP_3}$.

n	δ	η	$p_{\max} = 20$		$p_{\max} = 50$		$p_{\max} = 100$	
			avg	max	avg	max	avg	max
20	$\frac{1}{3}$	0.4	0.0055	0.0450	0.0057	0.0183	0.0098	0.0178
20	$\frac{1}{3}$	0.8	0.0025	0.0041	0.0058	0.0095	0.0106	0.0150
20	$\frac{1}{2}$	0.4	0.0020	0.0030	0.0046	0.0128	0.0183	0.2155
20	$\frac{1}{2}$	0.8	0.0021	0.0035	0.0043	0.0062	0.0079	0.0103
20	$\frac{2}{3}$	0.4	0.0015	0.0057	0.0025	0.0042	0.0053	0.0067
20	$\frac{2}{3}$	0.8	0.0014	0.0028	0.0034	0.0048	0.0054	0.0086
50	$\frac{1}{3}$	0.4	0.0590	0.0802	0.1705	0.2762	0.3950	0.5389
50	$\frac{1}{3}$	0.8	0.0757	0.1071	0.2290	0.2844	0.4802	0.6203
50	$\frac{1}{2}$	0.4	0.0442	0.0529	0.1100	0.1418	0.2313	0.3248
50	$\frac{1}{2}$	0.8	0.0512	0.0648	0.1435	0.1933	0.3222	0.4067
50	$\frac{2}{3}$	0.4	0.0252	0.0293	0.0599	0.0737	0.1193	0.1492
50	$\frac{2}{3}$	0.8	0.0287	0.0332	0.0682	0.0895	0.1455	0.1728
80	$\frac{1}{3}$	0.4	0.5925	0.7487	1.5261	1.7458	3.4211	5.7283
80	$\frac{1}{3}$	0.8	0.6379	0.8377	1.5512	1.9907	3.2992	4.5423
80	$\frac{1}{2}$	0.4	0.3530	0.4427	1.0609	1.1975	2.1062	2.6130
80	$\frac{1}{2}$	0.8	0.3534	0.4548	1.1035	1.4470	2.3446	3.3042
80	$\frac{2}{3}$	0.4	0.1764	0.2167	0.4676	0.7140	1.0303	1.3295
80	$\frac{2}{3}$	0.8	0.1632	0.1810	0.4392	0.5613	1.0856	1.4683
100	$\frac{1}{3}$	0.4	1.3943	1.6748	3.6191	4.7778	20.8650	128.7555
100	$\frac{1}{3}$	0.8	1.5780	1.8018	3.9061	4.5804	22.4305	160.2871
100	$\frac{1}{2}$	0.4	1.1196	1.4030	2.7046	3.0691	6.3306	16.1237
100	$\frac{1}{2}$	0.8	0.9295	1.1779	2.5127	3.2552	6.0045	12.7177
100	$\frac{2}{3}$	0.4	0.4326	0.5078	1.2432	1.7200	2.8508	3.6478
100	$\frac{2}{3}$	0.8	0.4312	0.5245	1.1629	1.4244	2.5620	3.0367
120	$\frac{1}{3}$	0.4	2.8138	3.4084	13.2120	18.4331	–	–
120	$\frac{1}{3}$	0.8	1.9996	2.6245	10.2410	17.7655	–	–
120	$\frac{1}{2}$	0.4	1.5391	1.8971	5.8844	12.6498	–	–
120	$\frac{1}{2}$	0.8	1.5609	2.2152	6.2663	13.7242	–	–
120	$\frac{2}{3}$	0.4	0.6451	0.8561	1.9487	2.8514	12.4595	15.2498
120	$\frac{2}{3}$	0.8	0.6638	0.8898	1.9389	3.6391	9.2060	11.4415

of indices k and l such that J_k^A is tardy and J_l^A is non-tardy, but $1 \le k \le v_A < l \le n_A$. We construct another solution ϕ' from ϕ by interchanging the positions of jobs J_k^A and J_l^A and their due date assignments as well. Since $p_k^A \le p_l^A$ and $w_k^A \ge w_l^A$, we have $C_k^A(\phi') = C_l^A(\phi) - p_l^A + p_k^A \le C_l^A(\phi) \le D_l^A(\phi) = D_k^A(\phi')$ and $C_j^X(\phi') \le C_j^X(\phi)$ for each $J_j^X \in \mathcal{J} \setminus \{J_l^A\}$. Hence, we have $\sum w_j^A U_j^A(\phi') = \sum w_j^A U_j^A(\phi) - w_k^A + w_l^A \le \sum w_j^A U_j^A(\phi)$ and $f^B(\phi') \le f^B(\phi)$. It follows that ϕ' is not worse than ϕ. By repeating the above process successively, we can get an optimal solution with the required property. □

In view of Lemma 4.1, it follows that in an optimal solution to $1|\mathrm{ADD}, \mathrm{Ant}(p_j^A, w_j^A)|\sum w_j^A U_j^A + \lambda f^B$, the set of non-tardy jobs can be determined in linear time, provided that the number of tardy jobs is given in advance. Combining this observation with the result of Lemma 3.4, we obtain the following remark.

Remark 4.1. There exists an optimal solution to $1|\mathrm{ADD}, \mathrm{Ant}(p_j^A, w_j^A)|\sum w_j^A U_j^A + \lambda f^B$ in which the A-jobs are executed in the SPT order.

Lemma 4.2. *There exists an optimal solution to* $1|\mathrm{ADD}, p_j^A = p^A|\sum Y_j^A + \lambda f^B$ *in which* $D_j^A = d_j^A$ *for* $j = 1, 2, \ldots, n_A$.

Proof. The result holds since all A-jobs have the same processing times. □

In view of Lemma 4.2, when one agent's scheduling criterion is the total late work and all its jobs have equal processing times, it follows that in an optimal solution the jth due date of this agent can be assigned to its jth job.

With respect to $1|\mathrm{ADD}, p_j^A = p^A|\sum w_j^A Y_j^A + \lambda f^B$, we assume that the A-jobs in $\mathcal{J}^A$ are indexed such that $w_1^A \ge w_2^A \ge \cdots \ge w_{n_A}^A$.

Lemma 4.3. *For* $1|\mathrm{ADD}, p_j^A = p^A|\sum w_j^A Y_j^A + \lambda f^B$, *if there are exactly* u_A *fully late A-jobs in an optimal solution, then there exists an optimal solution in which:*

(1) the jobs $J^A_{n_A-u_A+1}, J^A_{n_A-u_A+2}, \ldots, J^A_{n_A}$ *are fully late;*

(2) the due dates $d^A_{u_A+1}, d^A_{u_A+2}, \ldots, d^A_{n_A}$ *are assigned to the* $n_A - u_A$ *effective A-jobs according to the EDD rule such that* $D^A_{[j]} = d^A_{u_A+j}$ *for* $j = 1, 2, \ldots, n_A - u$.

Proof. Since the A-jobs are identical except their weights, in any optimal solution the fully late A-jobs must have the smallest weights, otherwise the solution is not optimal. Hence, Property (1) holds.

Combining the results of Lemma 3.1 and Property (1), we can assume that there exists an optimal solution ϕ in which the due dates in $D^A_{u_A}$ are assigned to the jobs in $\mathcal{J} \setminus \mathcal{J}^A_{n_A-u_A}$, and these jobs are sequenced after all effective jobs. If ϕ does not satisfy Property (2), we select the index g as small as possible such that $D^A_{[g]}(\phi) \ne d^A_{u_A+g}$ for some $g \in \{1, 2, \ldots, n_A - u_A\}$. The selection of g implies that $D^A_{[j]} = d^A_{u_A+j}$ for $j = 1, 2, \ldots, g-1$. Based on the above facts, the due date $d^A_{u_A+g}$ must be assigned to another effective A-job, say $J^A_{[h]}$, i.e., $D^A_{[h]}(\phi) = d^A_{u_A+g}$, where $g < h \le n_A - u_A$. We construct another solution ϕ' from ϕ by interchanging the due date assignments of jobs $J^A_{[g]}$ and $J^A_{[h]}$. From the selection of g, it follows that $D^A_{[g]}(\phi') = D^A_{[h]}(\phi) = d^A_{u_A+g}$ and $D^A_{[h]}(\phi') = D^A_{[g]}(\phi) \ge D^A_{[h]}(\phi)$. Recall that both $J^A_{[g]}$ and $J^A_{[h]}$ are effective in ϕ. Since $p^A_{[g]} = p^A_{[h]} = p^A$, we derive that $C^A_{[g]}(\phi') = C^A_{[g]}(\phi) \le C^A_{[h]}(\phi) - p^A < D^A_{[h]}(\phi) = d^A_{u_A+g} = D^A_{[g]}(\phi')$. Hence, if follows that $Y^A_{[g]}(\phi') = 0 = Y^A_{[g]}(\phi)$ and $Y^A_{[h]}(\phi') = \max\{0, C^A_{[h]}(\phi') - D^A_{[h]}(\phi')\} \le \max\{0, C^A_{[h]}(\phi) - D^A_{[h]}(\phi)\} = Y^A_{[h]}(\phi)$. This further indicates that $\sum w_j^A Y_j^A(\phi') \le \sum w_j^A Y_j^A(\phi)$. As $f^B(\phi') = f^B(\phi)$, so ϕ' is not worse than ϕ. By repeating the above process successively, we can get an optimal solution that satisfies Property (2). □

In view of Lemma 4.3, it follows that in an optimal solution to $1|\mathrm{ADD}, p_j^A = p^A|\sum w_j^A Y_j^A + \lambda f^B$, the set of non-fully late jobs can be determined in linear time, provided that the number of fully late jobs is given in advance.

4.1. *Problem* $1|\mathrm{ADD}, p_j^A = p_j^B = p|\sum Y_j^A + \lambda \sum Y_j^B$

In view of Lemma 4.2, we assume that $D_j^X = d_j^X$ for $X = A, B$, $j = 1, 2, \ldots, n_X$. Hence, $1|\mathrm{ADD}, p_j^A = p_j^B = p|\sum Y_j^A + \lambda \sum Y_j^B$ reduces to $1|p_j = p|\sum w_j Y_j$, where $w_j = 1$ if $J_j \in \mathcal{J}^A$, and $w_j = \lambda$ if $J_j \in \mathcal{J}^B$. Note that $1|p_j = p|\sum Y_j$ is solvable in $O(n \log n)$ time (Potts and Van Wassenhove, 1992), and $1|p_j = p|\sum w_j Y_j$ is solvable in $O(n^3)$ time (Hariri et al., 1995).

Theorem 4.1. *Problem* $1|\mathrm{ADD}, p_j^A = p_j^B = p|\sum Y_j^A + \lambda \sum Y_j^B$ *can be solved in* $O(n^3)$ *time and problem* $1|\mathrm{ADD}, p_j^A = p_j^B = p|\sum Y_j^A + \sum Y_j^B$ *can be solved in* $O(n \log n)$ *time.*

4.2. *Problem* $1|\mathrm{ADD}, p_j^A = p^A, p_j^B = p^B|\sum Y_j^A + \lambda \sum Y_j^B$

In view of Lemmas 3.1, 3.2 and 4.2, for $X = A, B$, we concentrate on the solutions (called *normal*) in which: (i) $D_j^X = d_j^X$ for $j = 1, 2, \ldots, n_X$; (ii) if there are u_X fully late X-jobs, then the jobs in $\mathcal{J}^X_{u_X}$ are fully late; and (iii) the effective X-jobs are executed according to the EDD sequence. However, due to the arbitrary sets $\mathcal{D}^A$ and $\mathcal{D}^B$, we cannot determine u_A and u_B in advance. To solve $1|\mathrm{ADD}, p_j^A = p^A, p_j^B = p^B|\sum Y_j^A + \lambda \sum Y_j^B$, we enumerate all potential pairs of u_A and u_B.

Assume that there are exactly u_A fully late A-jobs and u_B fully late B-jobs in an optimal solution, where $0 \le u_A \le n_A$ and $0 \le u_B \le n_B$. Set $Q(j_A, j_B) = (j_A - u_A)p^A + (j_B - u_B)p^B$ for each $j_A = u_A, u_A + 1, \ldots, n_A$ and $j_B = u_B, u_B + 1, \ldots, n_B$. Let the vector (j_A, j_B) denote a state standing for the scenario where the jobs in $\mathcal{J}^A_{u_A} \cup \mathcal{J}^B_{u_B}$ are all fully late, and the jobs in $\{J^A_{u_A+1}, J^A_{u_A+2}, \ldots, J^A_{j_A}\} \cup \{J^B_{u_B+1}, J^B_{u_B+2}, \ldots, J^B_{j_B}\}$ have been scheduled effective. Define $H(j_A, j_B)$ to be the optimal solution value for state (j_A, j_B). If there is no such solution, then we define $H(j_A, j_B) = +\infty$.

In an optimal normal solution to $1|\mathrm{ADD}, p_j^A = p^A, p_j^B = p^B|\sum Y_j^A + \lambda \sum Y_j^B$, the last scheduled job is either $J^A_{j_A}$ or $J^B_{j_B}$. In the former situation, we have $H(j_A, j_B) = H(j_A - 1, j_B) + \max\{0, Q(j_A, j_B) - d^A_{j_A}\}$, which happens only when $Q(j_A - 1, j_B) < d^A_{j_A}$. In the latter situation, we have $H(j_A, j_B) = H(j_A, j_B - 1) + \lambda \max\{0, Q(j_A, j_B) - d^B_{j_B}\}$, which happens only when $Q(j_A, j_B - 1) < d^B_{j_B}$.

Taking into account the preceding discussion, the following optimization algorithm (denoted $\mathbf{A_1}$) is introduced for $1|\mathrm{ADD}, p_j^A = p^A, p_j^B = p^B|\sum Y_j^A + \lambda \sum Y_j^B$.

Algorithm $\mathbf{A_1}$: $1|\mathrm{ADD}, p_j^A = p^A, p_j^B = p^B|\sum Y_j^A + \lambda \sum Y_j^B$

Step 1. Set $D_j^X = d_j^X$ for $X = A, B$ and $j = 1, 2, \ldots, n_X$.

Step 2. For each $X = A, B$ and $u_X = 0, 1, \ldots, n_X$, set $Q(j_A, j_B) = (j_A - u_A)p^A + (j_B - u_B)p^B$ for $j_A = u_A, u_A + 1, \ldots, n_A$ and $j_B = u_B, u_B + 1, \ldots, n_B$, do the following:

- **Initialization:** Set $H(u_A, u_B) = u_A p^A + \lambda u_B p_B$ and $H(j_A, j_B) = +\infty$ if $j_A < u_A$ or $j_B < u_B$.
- **Recursion:** For each $j_A = u_A, u_A + 1, \ldots, n_A$, $j_B = u_B, u_B + 1, \ldots, n_B$, and $j_A + j_B \ge u_A + u_B + 1$,

$$H(j_A, j_B) = \min \begin{cases} H(j_A - 1, j_B) + \max\{0, Q(j_A, j_B) - d^A_{j_A}\}, & \text{if } Q(j_A - 1, j_B) < d^A_{j_A}; \\ H(j_A, j_B - 1) + \lambda \max\{0, Q(j_A, j_B) - d^B_{j_B}\}, & \text{if } Q(j_A, j_B - 1) < d^B_{j_B}; \\ +\infty, & \text{otherwise.} \end{cases}$$

- **Result:** Set $H^*(u_A, u_B) = H(n_A, n_B)$.

Step 3. The optimum solution value is determined by $H^* = \min\{H^*(u_A, u_B) : u_A = 0, 1, \ldots, n_A; u_B = 0, 1, \ldots, n_B\}$.

Theorem 4.2. *Problem* $1|\text{ADD}, p_j^A = p^A, p_j^B = p^B| \sum Y_j^A + \lambda \sum Y_j^B$ *can be solved by Algorithm* $\mathbf{A_1}$ *in* $O(n_A^2 n_B^2)$ *time.*

Proof. The correctness of Algorithm $\mathbf{A_1}$ stems from the properties established in Lemmas 3.1, 3.2 and 4.2 and the above discussion. Note that for each pair of (u_A, u_B), the recursion in Step 2 costs $O((n_A - u_A)(n_B - u_B))$ time. Since $O(n_A n_B)$ pairs of (u_A, u_B) have to be enumerated, the time complexity of $\mathbf{A_1}$ is $O(n_A^2 n_B^2)$. □

Remark 4.2. When the weights of jobs are arbitrary, the computational complexity of problem $1|\text{ADD}, p_j^A = p_j^B = p| \sum w_j^A Y_j^A + \lambda \sum w_j^B Y_j^B$ is still open.

4.3. *Problem* $1|\text{ADD}, p_j^B = p^B| \sum T_j^A + \lambda \sum Y_j^B$

In view of Lemmas 3.1, 3.2, 3.3 and 4.2, we concentrate on the solutions (called *normal*) in which: (i) $D_j^B = d_j^B$ for $j = 1, 2, \ldots, n_B$; (ii) the A-jobs are executed in line with the SPT-EDD principle; and (iii) if there are u_B fully late B-jobs, then the jobs in $\mathcal{J}_{u_B}^B$ are fully late; and (iv) the effective B-jobs are executed according to the EDD rule. To solve $1|\text{ADD}, p_j^B = p^B| \sum T_j^A + \lambda \sum Y_j^B$, we enumerate all potential values of u_B.

Assume that there are exactly u_B fully late B-jobs in an optimal solution, where $0 \le u_B \le n_B$. Set $R(j_A, j_B) = P_{j_A}^A + (j_B - u_B)p^B$ for each $j_A = 0, 1, \ldots, n_A$ and $j_B = u_B, u_B + 1, \ldots, n_B$. Let the vector (j_A, j_B) denote a state standing for the scenario where the jobs in $\mathcal{J}_{j_A}^A \cup \mathcal{J}_{j_B}^B$ have been scheduled, the jobs in $\mathcal{J}_{u_B}^B$ are all fully late, and the jobs in $\mathcal{J}_{j_B}^B \setminus \mathcal{J}_{u_B}^B$ have been scheduled effective. Define $H(j_A, j_B)$ to be the optimal solution value for state (j_A, j_B). If there is no such solution, then we define $H(j_A, j_B) = +\infty$.

In an optimal normal solution to $1|\text{ADD}, p_j^B = p^B| \sum T_j^A + \lambda \sum Y_j^B$, the last scheduled job is either $J_{j_A}^A$ or $J_{j_B}^B$. In the former situation, we have $H(j_A, j_B) = H(j_A - 1, j_B) + T_{j_A}^A$, where $T_{j_A}^A = \max\{0, R(j_A, j_B) - d_{j_A}^A\}$. In the latter situation, we have $H(j_A, j_B) = H(j_A, j_B - 1) + \lambda Y_{j_B}^B$, where $Y_{j_B}^B = \min\{p^B, \max\{0, R(j_A, j_B) - d_{j_B}^B\}\} = \max\{0, R(j_A, j_B) - d_{j_B}^B\}$. Note that this happens only when $R(j_A, j_B - 1) < d_{j_B}^B$.

Taking into account the preceding discussion, the following optimization algorithm (denoted $\mathbf{A_2}$) is introduced for $1|\text{ADD}, p_j^B = p^B| \sum T_j^A + \lambda \sum Y_j^B$.

Algorithm $\mathbf{A_2}$: $1|\text{ADD}, p_j^B = p^B| \sum T_j^A + \lambda \sum Y_j^B$

Step 1. Set $D_j^X = d_j^X$ for $X = A, B$ and $j = 1, 2, \ldots, n_X$.

Step 2. For each $u_B = 0, 1, \ldots, n_B$, set $R(j_A, j_B) = P_{j_A}^A + (j_B - u_B)p^B$ for $j_A = 0, 1, \ldots, n_A$, do the following:

- **Initialization:** Set $H(0, u_B) = \lambda u_B p_B$ and $H(j_A, j_B) = +\infty$ if $j_A < 0$ or $j_B < u_B$.
- **Recursion:** For each $j_A = 0, 1, \ldots, n_A$, $j_B = u_B, u_B + 1, \ldots, n_B$, and $j_A + j_B \ge u_B + 1$,

$$H(j_A, j_B) = \min \begin{cases} H(j_A - 1, j_B) + \max\{0, R(j_A, j_B) - d_{j_A}^A\}; \\ H(j_A, j_B - 1) + \lambda \max\{0, R(j_A, j_B) - d_{j_B}^B\}, \quad \text{if } R(j_A, j_B - 1) < d_{j_B}^B. \end{cases}$$

- **Result:** Set $H^*(u_B) = H(n_A, n_B)$.

Step 3. The optimum solution value is determined by $H^* = \min\{H^*(u_B) : u_B = 0, 1, \ldots, n_B\}$.

Theorem 4.3. *Problem* $1|\text{ADD}, p_j^B = p^B| \sum T_j^A + \lambda \sum Y_j^B$ *can be solved by Algorithm* $\mathbf{A_2}$ *in* $O(n_A n_B^2)$ *time.*

Proof. The correctness of Algorithm $\mathbf{A_2}$ stems from the properties established in Lemmas 3.1, 3.2, 3.3 and 4.2 and the above discussion. For each u_B, the recursion in Step 2 costs $O(n_A(n_B - u_B))$ time. Thus, the time complexity of $\mathbf{A_2}$ is $O(n_A n_B^2)$. □

4.4. *Problem* $1|\text{ADD}, \text{Ant}(p_j^A, w_j^A), p_j^B = p^B| \sum w_j^A U_j^A + \lambda \sum w_j^B Y_j^B$

The next lemma will help us determine which effective job to schedule in the last position, given the number u_A of tardy A-jobs and the number u_B of fully late B-jobs in advance.

Lemma 4.4. *Consider a feasible instance of* $1|\text{ADD}, \text{Ant}(p_j^A, w_j^A), p_j^B = p^B| \sum w_j^A U_j^A + \lambda \sum w_j^B Y_j^B$ *with* u_A *tardy* A*-jobs and* u_B *fully late* B*-jobs in an optimal solution, and let* $v_A = n_A - u_A$, $v_B = n_B - u_B$ *and* $\Delta = P_{v_A}^A + v_B p^B$. *There exists an optimal solution in which* $J_{v_A}^A$ *is executed in the interval* $[\Delta - p_{v_A}^A, \Delta]$ *if* $d_{n_A}^A \ge \Delta$, *and a* B*-job must be executed in the interval* $[\Delta - p^B, \Delta]$ *if* $d_{n_A}^A < \Delta$.

Proof. Let ϕ be an optimal solution with u_A tardy A-jobs and u_B fully late B-jobs that satisfies the properties established in Lemmas 3.1, 3.4, 4.1 and 4.3. In ϕ, we can assume that: (i) the jobs in $\mathcal{J}^A \setminus \mathcal{J}_{v_A}^A$ are tardy, the due dates in $D_{u_A}^A$ are assigned to these tardy A-jobs, and $D_{v_A+j}^A(\phi) = d_j^A$ for $j = 1, 2, \ldots, u_A$; (ii) the v_A effective A-jobs are executed in line with the SPT sequence, and $D_j^A(\phi) = d_{u_A+j}^A$ for $j = 1, 2, \ldots, v_A$; (iii) the jobs in $\mathcal{J}^B \setminus \mathcal{J}_{v_B}^B$ are fully late, the due dates in $D_{u_B}^B$ are assigned to these fully late jobs, and $D_{v_B+j}^B(\phi) = d_j^B$ for $j = 1, 2, \ldots, u_B$; (iv) the effective B-jobs are executed in line with the sequence $(J_{[1]}^B, J_{[2]}^B, \ldots, J_{[v_B]}^B)$, and $D_{[j]}^B(\phi) = d_{u_B+j}^B$ for $j = 1, 2, \ldots, v_B$; and (v) the jobs in $(\mathcal{J}^A \setminus \mathcal{J}_{v_A}^A) \cup (\mathcal{J}^B \setminus \mathcal{J}_{v_B}^B)$ are executed in the interval $[\Delta, P]$. This further implies that the jobs in $\mathcal{J}_{v_A}^A \cup \mathcal{J}_{v_B}^B$ are all effective and either $J_{v_A}^A$ or a B-job is completed at time Δ.

As a consequence, if $d_{n_A}^A < \Delta$, then no A-job can be completed at time Δ in ϕ, so a B-job must be scheduled in $[\Delta - p^B, \Delta]$. Assume that $d_{n_A}^A \ge \Delta$ and $J_{v_A}^A$ is not executed in the interval $[\Delta - p_{v_A}^A, \Delta]$ in ϕ. We construct another solution ϕ' from ϕ by scheduling $J_{v_A}^A$ in $[\Delta - p_{v_A}^A, \Delta]$. Clearly, $J_{v_A}^A$ is still effective and the completion times of all other jobs do not increase in ϕ'. Hence, ϕ' is an optimal solution that satisfies the required property. □

In view of Lemmas 3.1, 3.4, 4.1, 4.3 and 4.4, the following algorithm (denoted $\mathbf{A_3}$) is provided to solve $1|\text{ADD}, \text{Ant}(p_j^A, w_j^A), p_j^B = p^B| \sum w_j^A U_j^A + \lambda \sum w_j^B Y_j^B$. In $\mathbf{A_3}$, we enumerate all potential pairs of u_A and u_B, which denote the number of tardy A-jobs and fully late B-jobs, respectively, in the final solution. For a given feasible pair (u_A, u_B), the tardy A-jobs and fully late B-jobs are executed in the interval $[P_{v_A}^A + v_B p^B, P]$ arbitrarily, while the due date assignments are given in advance. Specifically, Δ signifies the sum of all unscheduled jobs' processing times, ρ signifies the sequence of scheduled effective jobs, l signifies the number of unscheduled jobs, g and h signify respectively the number of unscheduled A-jobs and B-jobs, where $g + h = l$. Algorithm $\mathbf{A_3}$ schedules the effective jobs backwards. At each iteration, by Lemma 4.4, if $D_g^A = d_{u_A+g}^A \ge \Delta$, job J_g^A is scheduled in the interval $[\Delta - p_g^A, \Delta]$, otherwise a B-job should be sequenced in $[\Delta - p^B, \Delta]$, and the late work $Y_{[h]}^B$ of the hth completed B-job is recorded. When all $Y_{[j]}^B$, $h = 1, 2, \ldots, v_B$, are determined, the objective function value of this case can be formulated as

$$H(u_A, u_B) = \sum_{j=v_A+1}^{n_A} w_j^A + \lambda p^B \sum_{j=v_B+1}^{n_B} w_j^B + \lambda \sum_{h=1}^{v_B} w_{[h]}^B Y_{[h]}^B,$$

where the first two terms in $H(u_A, u_B)$ are both constants, and $([1], [2], \ldots, [v_B])$ is a permutation of $(1, 2, \ldots, v_B)$. Define $x_{j,h} = 1$ if J_j^B is the hth completed job, and $x_{j,h} = 0$ otherwise, $j, h = 1, 2, \ldots, v_B$. Define $c_{j,h} = w_j^B Y_{[h]}^B$ for $j, h = 1, 2, \ldots, v_B$. Then the third term in $H(u_A, u_B)$

reduces to the following linear assignment (LP) problem:

$$\text{Min} \sum_{j=1}^{v_B}\sum_{h=1}^{v_B} c_{j,h}x_{j,h}$$

$$\text{s.t.} \sum_{h=1}^{v_B} x_{j,h} = 1, \quad j = 1,2,\dots,v_B,$$

$$\sum_{j=1}^{v_B} x_{j,h} = 1, \quad h = 1,2,\dots,v_B,$$

$$x_{j,h} \in \{0,1\}, \quad j = 1,2,\dots,v_B; \quad h = 1,2,\dots,v_B.$$

Note that the cost matrix $(c_{j,h})_{v_B\times v_B}$ can be formulated as the product of two vectors $(Y^B_{[1]}, Y^B_{[2]}, \dots, Y^B_{[v_B]})^T$ and $(w^B_1, w^B_2, \dots, w^B_{v_B})$, i.e., $(c_{j,h})_{v_B\times v_B} = (Y^B_{[1]}, Y^B_{[2]}, \dots, Y^B_{[v_B]})^T \times (w^B_1, w^B_2, \dots, w^B_{v_B})$. Then the LP problem can be solved by matching the effective B-jobs' weights to the given late work according to the well-known Hungarian method (Kuhn, 1955). The details of $\mathbf{A_3}$ are given below.

Algorithm $\mathbf{A_3}$: $1|\text{ADD}, \text{Ant}(p^A_j, w^A_j), p^B_j = p^B|\sum w^A_j U^A_j + \lambda\sum w^B_j Y^B_j$

Step 1. Renumber the A-jobs such that $p^A_1 \le p^A_2 \le \cdots \le p^A_{n_A}$ and $w^A_1 \ge w^A_2 \ge \cdots \ge w^A_{n_A}$, and renumber the B-jobs such that $w^B_1 \ge w^B_2 \ge \cdots \ge w^B_{n_B}$.

Step 2. For each $X = A, B$ and $u_X = 0,1,\dots,n_X$, set $v_X = n_X - u_X$, do the following:

(1) Set $D^A_j = d^A_{u_A+j}$ for $j = 1,2,\dots,v_A$, $D^A_{v_A+j} = d^A_j$ for $j = 1,2,\dots,u_A$, $D^B_{[j]} = d^B_{u_B+j}$ for $j = 1,2,\dots,v_B$, $D^B_{v_B+j} = d^B_j$ for $j = 1,2,\dots,u_B$, $\Delta := P^A_{v_A} + v_B p^B$, $\rho := ()$, $l := v_A + v_B$, $g := v_A$, $h := v_B$.

(2) If $g \ge 1$, then go to (3); otherwise go to (4).

(3) If $D^A_g \ge \Delta$, set $\Delta := \Delta - p^A_g$, $\rho := (J^A_g) \parallel \rho$, $l := l-1$, $g := g-1$, then go to (5); otherwise go to (4).

(4) If $h \ge 1$ and $\Delta - p^B < D^B_{[h]}$, set $Y^B_{[h]} = \max\{0, \Delta - D^B_{[h]}\}$, $\Delta := \Delta - p^B$, $\rho := (J^B_{[h]}) \parallel \rho$, $l := l-1$, $h := h-1$, then go to (5); otherwise set $H(u_A, u_B) = +\infty$ (no feasible solution exists) and stop.

(5) If $l \ge 1$, then go to (2); otherwise go to (6).

(6) Sort the late works of the v_B positions such that $Y^B_{([i_1])} \le Y^B_{([i_2])} \le \cdots \le Y^B_{([i_{v_B}])}$, then set $H(u_A, u_B) = \sum_{j=v_A+1}^{n_A} w^A_j + \lambda(p^B\sum_{j=v_B+1}^{n_B} w^B_j + \sum_{j=1}^{v_B} w^B_j Y^B_{[i_j]})$, $J^B_j = J^B_{[i_j]}$, $D^B_j = D^B_{[i_j]} = d^B_{u_B+i_j}$ for $j = 1,2,\dots,v_B$.

Step 3. The optimum solution value is determined by $H^* = \min\{H(u_A, u_B) : u_A = 0,1,\dots,n_A; u_B = 0,1,\dots,n_B\}$.

Theorem 4.4. *Problem* $1|\text{ADD}, \text{Ant}(p^A_j, w^A_j), p^B_j = p^B|\sum w^A_j U^A_j + \lambda\sum w^B_j Y^B_j$ *can be solved by Algorithm* $\mathbf{A_3}$ *in* $O(n_A n_B(n + n_B\log n_B))$ *time.*

Proof. The correctness of Algorithm $\mathbf{A_3}$ stems from the properties established in Lemmas 3.1, 3.4, 4.1, 4.3, 4.4 and the above discussion. Step 1 takes $O(n_A\log n_A + n_B\log n_B)$ time for renumbering. In Step 2, for each given pair (u_A, u_B), Step (2.1) costs linear time, Steps (2.2)-(2.5) are executed at most $v_A + v_B \le n$ times, and each execution takes constant time, Step (2.6) costs $O(v_B\log v_B)$ time. Since $u_A \le n_A$ and $u_B \le n_B$, Step 2 takes $O(n_A n_B(n + n_B\log n_B))$ time. Step 3 takes $O(n_A n_B)$ time. Thus, the time complexity of $\mathbf{A_3}$ is $O(n_A n_B(n + n_B\log n_B))$. □

Corollary 4.1. *Problem* $1|\text{ADD}, p^B_j = p^B|\sum U^A_j + \lambda\sum w^B_j Y^B_j$ *can be solved in* $O(n_A n_B(n + n_B\log n_B))$ *time.*

Corollary 4.2. *Problem* $1|\text{ADD}, p^A_j = p^A, p^B_j = p^B|\sum w^A_j U^A_j + \lambda\sum w^B_j Y^B_j$ *can be solved in* $O(n_A n_B(n + n_B\log n_B))$ *time.*

Table 4
Job data of the instance.

J^X_j	J^A_1	J^A_2	J^B_1	J^B_2	J^B_3
p^X_j	5	6	4	4	4
w^X_j	10	5	4	3	2

Example 4.1. Consider a 5-job instance of $1|\text{ADD}, \text{Ant}(p^A_j, w^A_j), p^B_j = p^B|\sum w^A_j U^A_j + \lambda\sum w^B_j Y^B_j$, in which $\mathcal{J}^A = \{J^A_1, J^A_2\}$, $\mathcal{J}^B = \{J^B_1, J^B_2, J^B_3\}$, $\mathcal{D}^A = \{6, 11\}$, $\mathcal{D}^B = \{3, 7, 12\}$, the weighting factor is $\lambda = 1$, and the job parameters are given in Table 4.

By applying Algorithm $\mathbf{A_3}$, we get an optimal solution ϕ in which the job processing sequence is given by $J^B_1 \to J^A_1 \to J^B_2 \to J^A_2 \to J^B_3$, and the due date assignment strategy is given by $D^A_1(\phi) = 11$, $D^A_2(\phi) = 6$, $D^B_1(\phi) = 7$, $D^B_2(\phi) = 12$ and $D^B_3(\phi) = 3$. Then we have $U^A_1(\phi) = 0$, $U^A_2(\phi) = 1$, $Y^B_1(\phi) = 0$, $Y^B_2(\phi) = 1$ and $Y^B_3(\phi) = 4$. This implies that J^A_1, J^B_1 and J^B_2 are effective jobs, J^A_2 is a tardy job, and J^B_3 is a fully late job. Hence, the optimal solution value is equal to $\sum w^A_j U^A_j + \lambda\sum w^B_j Y^B_j = 5+2\times4+0+3 = 16$. The detailed implementation of $\mathbf{A_3}$ is illustrated in Appendix B.

4.5. Problem $1|\text{ADD}, \text{Ant}(p^A_j, w^A_j)|\sum w^A_j U^A_j + \lambda\sum T^B_j$

The next lemma will help us determine which effective job to schedule in the last position, given the number u_A of tardy A-jobs in advance.

Lemma 4.5. *Consider a feasible instance of* $1|\text{ADD}, \text{Ant}(p^A_j, w^A_j)|\sum w^A_j U^A_j + \lambda\sum T^B_j$ *with* u_A *tardy* A*-jobs in an optimal solution, and let* $v_A = n_A - u_A$ *and* $\Delta = P^A_{v_A} + P^B_{n_B}$. *There exists an optimal solution in which* $J^A_{v_A}$ *is executed in the interval* $[\Delta - p^A_{v_A}, \Delta]$ *if* $d^A_{n_A} \ge \Delta$, *and* $J^B_{n_B}$ *is executed in the interval* $[\Delta - p^B_{n_B}, \Delta]$ *if* $d^A_{n_A} < \Delta$.

Proof. Let ϕ be an optimal solution with u_A tardy A-jobs that satisfies the properties established in Lemmas 3.3, 3.4 and 4.1. In ϕ, we can assume that: (i) the jobs in $\mathcal{J}^A \setminus \mathcal{J}^A_{v_A}$ are tardy, the due dates in $D^A_{u_A}$ are assigned to these tardy A-jobs, and $D^A_{v_A+j}(\phi) = d^A_j$ for $j = 1,2,\dots,u_A$; (ii) the v_A effective A-jobs are executed in line with the SPT sequence, and $D^A_j(\phi) = d^A_{u_A+j}$ for $j = 1,2,\dots,v_A$; (iii) the B-jobs are executed in line with the SPT-EDD principle; and (iv) the jobs in $\mathcal{J}^A \setminus \mathcal{J}^A_{v_A}$ are executed in the interval $[\Delta, P]$. This further implies that the jobs in $\mathcal{J}^A_{v_A} \cup \mathcal{J}^B_{n_B}$ are all effective and either $J^A_{v_A}$ or $J^B_{n_B}$ is completed at time Δ.

As a consequence, if $d^A_{n_A} < \Delta$, then no A-job can be completed at time Δ in ϕ, so $J^B_{n_B}$ is executed in the interval $[\Delta - p^B_{n_B}, \Delta]$. Assume that $d^A_{n_A} \ge \Delta$ and job $J^A_{v_A}$ is not executed in $[\Delta - p^A_{v_A}, \Delta]$ in ϕ. We construct another solution ϕ' from ϕ by scheduling $J^A_{v_A}$ in $[\Delta - p^A_{v_A}, \Delta]$. Clearly, $J^A_{v_A}$ is still effective and the completion times of all other jobs do not increase in ϕ'. Hence, ϕ' is an optimal solution that satisfies the required property. □

In view of Lemmas 3.3, 3.4, 4.1 and 4.5, the following optimization algorithm (denoted $\mathbf{A_4}$) is introduced for $1|\text{ADD}, \text{Ant}(p^A_j, w^A_j)|\sum w^A_j U^A_j + \lambda\sum T^B_j$.

Algorithm $\mathbf{A_4}$: $1|\text{ADD}, \text{Ant}(p^A_j, w^A_j)|\sum w^A_j U^A_j + \lambda\sum T^B_j$

Step 1. Renumber the A-jobs such that $p^A_1 \le p^A_2 \le \cdots \le p^A_{n_A}$ and $w^A_1 \ge w^A_2 \ge \cdots \ge w^A_{n_A}$, and set $D^B_j = d^B_j$ for $j = 1,2,\dots,n_B$.

Step 2. For each $u_A = 0,1,\dots,n_A$, set $v_A = n_A - u_A$, do the following:

(1) Set $D^A_j = d^A_{u_A+j}$ for $j = 1,2,\dots,v_A$, $D^A_{v_A+j} = d^A_j$ for $j = 1,2,\dots,u_A$, $H(u_A) := \sum_{j=v_A+1}^{n_A} w^A_j$, $\Delta := P^A_{v_A} + P^B_{n_B}$, $\rho := ()$, $l := v_A + n_B$, $g := v_A$, $h := n_B$.

Table 5
New results obtained in this paper.

Problem	Complexity	Reference
$1\|\text{ADD}\|\sum Y_j^A + \lambda\sum Y_j^B$	binary $\mathcal{NP}$-hard, $O(n_A^2 n_B^2 P)$	Section 3.1
$1\|\text{ADD}\|\sum Y_j$	improved complexity to $O(n^2 P)$	Section 3.1
$1\|\text{ADD}\|\sum w_j^A U_j^A + \lambda\sum Y_j^B$	binary $\mathcal{NP}$-hard, $O(n_A^2 n_B^2 P)$	Section 3.2
$1\|\text{ADD}\|\sum T_j^A + \lambda\sum Y_j^B$	binary $\mathcal{NP}$-hard, $O(n_A n_B^2 P)$	Section 3.3
$1\|\text{ADD}, p_j^A = p_j^B = p\|\sum Y_j^A + \lambda\sum Y_j^B$	$O(n^3)$	Section 4.1
$1\|\text{ADD}, p_j^A = p_j^B = p\|\sum Y_j^A + \sum Y_j^B$	$O(n\log n)$	Section 4.1
$1\|\text{ADD}, p_j^A = p^A, p_j^B = p^B\|\sum Y_j^A + \lambda\sum Y_j^B$	$O(n_A^2 n_B^2)$	Section 4.2
$1\|\text{ADD}, p_j^B = p^B\|\sum T_j^A + \lambda\sum Y_j^B$	$O(n_A n_B^2)$	Section 4.3
$1\|\text{ADD}, \text{Ant}(p_j^A, w_j^A), p_j^B = p^B\|\sum w_j^A U_j^A + \lambda\sum w_j^B Y_j^B$	$O(n_A n_B(n + n_B\log n_B))$	Section 4.4
$1\|\text{ADD}, p_j^B = p^B\|\sum U_j^A + \lambda\sum w_j^B Y_j^B$	$O(n_A n_B(n + n_B\log n_B))$	Section 4.4
$1\|\text{ADD}, p_j^A = p^A, p_j^B = p^B\|\sum w_j^A U_j^A + \lambda\sum w_j^B Y_j^B$	$O(n_A n_B(n + n_B\log n_B))$	Section 4.4
$1\|\text{ADD}, \text{Ant}(p_j^A, w_j^A)\|\sum w_j^A U_j^A + \lambda\sum T_j^B$	$O(nn_A)$	Section 4.5
$1\|\text{ADD}\|\sum U_j^A + \lambda\sum T_j^B$	$O(nn_A)$	Section 4.5
$1\|\text{ADD}, p_j^A = p^A\|\sum w_j^A U_j^A + \lambda\sum T_j^B$	$O(nn_A)$	Section 4.5

(2) If $g \geq 1$, then go to (3); otherwise go to (4).

(3) If $D_g^A \leq \Delta$, set $\Delta := \Delta - p_g^A$, $\rho := (J_g^A) \parallel \rho$, $l := l-1$, $g := g-1$, then go to (5); otherwise go to (4).

(4) If $h \geq 1$, set $H(u_A) := H(u_A) + \lambda\max\{0, \Delta - D_h^B\}$, $\Delta := \Delta - p_h^B$, $\rho := (J_h^B) \parallel \rho$, $l := l-1$, $h := h-1$, then go to (5); otherwise set $H(u_A) = +\infty$ (no feasible solution exists) and stop.

(5) If $l \geq 1$, then go to (2); otherwise stop.

Step 3. The optimum solution value is determined by $H^* = \min\{H(u_A) : u_A = 0, 1, \ldots, n_A\}$.

Theorem 4.5. *Problem* $1|\text{ADD}, \text{Ant}(p_j^A, w_j^A)|\sum w_j^A U_j^A + \lambda\sum T_j^B$ *can be solved by Algorithm* $\mathbf{A_4}$ *in* $O(nn_A)$ *time.*

Proof. The correctness of Algorithm $\mathbf{A_4}$ stems from the properties established in Lemmas 3.3, 3.4, 4.1 and 4.5 and the above discussion. Step 1 costs $O(n_A\log n_A + n_B\log n_B)$ time for renumbering. In Step 2, for each given value u_A, Steps (2.1)-(2.5) can be implemented in $O(n)$ time. In Step 3, it costs $O(n_A)$ time for computing the optimal solution value. Since u_A is enumerated $O(n_A)$ times, the time complexity of $\mathbf{A_4}$ is $O(nn_A)$. □

Corollary 4.3. *Problem* $1|\text{ADD}|\sum U_j^A + \lambda\sum T_j^B$ *can be solved in* $O(nn_A)$ *time.*

Corollary 4.4. *Problem* $1|\text{ADD}, p_j^A = p^A|\sum w_j^A U_j^A + \lambda\sum T_j^B$ *can be solved in* $O(nn_A)$ *time.*

5. Conclusions

We studied scheduling problems involving assignable due dates and two competing agents on a single machine. The solution to these problems comprises a due date assignment and a feasible schedule. The goal is to find a solution that minimizes the weighted sum of the scheduling criteria for both agents. Utilizing the structural properties of optimal solutions established in this paper, we proposed pseudo-polynomial-time DP algorithms for three basic models $1|\text{ADD}|\sum Y_j^A + \lambda\sum Y_j^B$, $1|\text{ADD}|\sum w_j^A U_j^A + \lambda\sum Y_j^B$ and $1|\text{ADD}|\sum T_j^A + \lambda\sum Y_j^B$, and conducted numerical tests to evaluate their performance. Additionally, we investigated special cases where the processing times for one or both agents are identical, as well as scenarios where the processing times and weights of the A-jobs are antithetical. For these special cases, we constructed polynomial-time algorithms. The new results are summarized in Table 5.

Our research has primarily focused on the theoretical studies establishing the complexity status of the basic single-machine models, which appeared to be binary NP-hard. To prove their pseudo-polynomial time solvability, we proposed a few DP algorithms. While these algorithms are of theoretical significance, computational experiments confirmed that they can also solve small- to medium-sized instances efficiently (taking into account the classification used in the related literature). The methods which allowed us excluding unary NP-hardness of studied models, appeared to be practically applicable, but they have time and memory limitations. Hence, the algorithmic studies, i.e. the research on other effective solution approaches, would be the next natural step after the presented complexity studies. The unique characteristics of the problem, including assignable due dates, competing agents, and late work-related criteria, present fundamental challenges for constructing exact algorithms, such as mixed-integer linear programming (MILP) formulations and branch-and-bound methods. Comparing the time efficiency of such approaches with DP methods developed in the presented research would be an interesting topic of the future research. Moreover, approximation algorithms with performance guarantees, as well as heuristic and meta-heuristic techniques, could be explored to efficiently handle larger instances while ensuring solution quality and computational efficiency. Another promising direction involves extending the current model to more general scheduling models with multiple agents and/or multiple machines (e.g., identical/uniform/unrelated parallel machines, as well as flow/open/job shops).

CRediT authorship contribution statement

Shi-Sheng Li: Writing – original draft, Software, Methodology, Investigation, Funding acquisition, Formal analysis. **Yao-Wen Sang:** Writing – review & editing, Methodology, Investigation, Formal analysis. **Ren-Xia Chen:** Writing – review & editing, Software, Methodology, Conceptualization. **Malgorzata Sterna:** Writing – review & editing, Validation, Supervision. **Tamas Kis:** Writing – review & editing, Supervision, Methodology, Formal analysis.

Acknowledgments

We would like to thank the Editor, Associate Editor and four anonymous reviewers for their insightful comments, which have contributed to improve the presentation of this research. This work was supported by the National Natural Science Foundation of China [grant numbers 12401427, 12471305], the Key Research Projects of Henan Higher Education Institutions [grant number 24A110014], the Fundamental Research Funds for the Central Universities of China [grant number NS2024048], the National Research, Development and Innovation Office of Hungary [grant number TKP2021-NKTA-01], and the statutory funds of Poznan University of Technology (Poland).

Appendix A

The detailed state transitions of Algorithm $\mathbf{DP_1}$ for Example 3.1 is illustrated, where only the values with $G(j_A, j_B, s, k_A, k_B) \neq +\infty$ are

recorded. Note that $n_A = n_B = 2$, $P = 14$, $P_1^A = 2$, $P_2^A = 6$, $P_1^B = 3$ and $P_2^B = 8$.

• When $j_A = 2$, $j_B = 3$, $s = 0, 1, \ldots, 10$ (due to $P_1^A + P_2^B = 10$), $k_A = 0, 1$ and $k_B = 0$, we have

$$G(2,3,s,k_A,0) = \begin{cases} 4, & \text{if } k_A = 1; \\ \max\{0, s+4-d_2^A\}, & \text{if } (s \le d_2^A - 1) \wedge (k_A = 0). \end{cases}$$

Thus, $G(2,3,s,0,0) = 0$ for $s = 0, 1, \ldots, 6$; $G(2,3,s,0,0) = s-6$ for $s = 7, 8, 9$;

$G(2,3,s,1,0) = 4$ for $s = 0, 1, \ldots, 10$.

• When $j_A = 3$, $j_B = 2$, $s = 0, 1, \ldots, 9$ (due to $P_2^A + P_1^B = 9$), $k_A = 0$ and $k_B = 0, 1$, we have

$$G(3,2,s,0,k_B) = \begin{cases} 5, & \text{if } k_B = 1; \\ \max\{0, s+5-d_2^B\}, & \text{if } (s \le d_2^B - 1) \wedge (k_B = 0). \end{cases}$$

Thus, $G(3,2,s,0,0) = 0$ for $s = 0, 1, 2$; $G(3,2,s,0,0) = s-2$ for $s = 3, 4, 5, 6$;

$G(3,2,s,0,1) = 5$ for $s = 0, 1, \ldots, 9$.

• When $j_A = 3$, $j_B = 1$, $s = 0, 1, \ldots, 6$ (due to $P_2^A = 6$), $k_A = 0$, $k_B = 0, 1, 2$, we have

$$G(3,1,s,0,k_B) = \min \begin{cases} G(3,2,s+3,0,k_B) + \max\{0, s+3-d_{1+k_B}^B\}, & \text{if } s \le d_{1+k_B}^B - 1; \\ G(3,2,s,0,k_B-1) + 3. \end{cases}$$

Thus, $G(3,1,0,0,0) = 1$; $G(3,1,1,0,0) = 2$; $G(3,1,2,0,0) = 4$; $G(3,1,3,0,0) = 6$;

$G(3,1,s,0,1) = 3$ for $s = 0, 1, 2$; $G(3,1,s,0,1) = s+1$ for $s = 3, 4, 5, 6$;

$G(3,1,s,0,2) = 8$ for $s = 0, 1, 2, \ldots, 6$.

• When $j_A = 2$, $j_B = 2$, $s = 0, 1, \ldots, 5$ (due to $P_1^A + P_1^B = 5$), $k_A = 0, 1$, $k_B = 0, 1$, we have

$$G(2,2,s,k_A,k_B) = \min \begin{cases} G(3,2,s+4,k_A,k_B) + \max\{0, s+4-d_{2+k_A}^A\}, & \text{if } s \le d_{2+k_A}^A - 1; \\ G(3,2,s,k_A-1,k_B) + 4; \\ G(2,3,s+5,k_A,k_B) + \max\{0, s+5-d_{2+k_B}^B\}, & \text{if } s \le d_{2+k_B}^B - 1; \\ G(2,3,s,k_A,k_B-1) + 5. \end{cases}$$

Thus, $G(2,2,s,0,0) = 0$ for $s = 0, 1$; $G(2,2,s,0,0) = 2s-3$ for $s = 2, 3, 4$;

$G(2,2,s,0,1) = 5$ for $s = 0, 1, \ldots, 5$;

$G(2,2,s,1,0) = 4$ for $s = 0, 1, 2$; $G(2,2,s,0,1) = s+2$ for $s = 3, 4, 5$;

$G(2,2,s,1,1) = 9$ for $s = 0, 1, \ldots, 5$.

• When $j_A = 2$, $j_B = 1$, $s = 0, 1, 2$ (due to $P_1^A = 2$), $k_A = 0, 1$, $k_B = 0, 1, 2$, we have

$$G(2,1,s,k_A,k_B) = \min \begin{cases} G(3,1,s+4,k_A,k_B) + \max\{0, s+4-d_{2+k_A}^A\}, & \text{if } s \le d_{2+k_A}^A - 1; \\ G(3,1,s,k_A-1,k_B) + 4; \\ G(2,2,s+3,k_A,k_B) + \max\{0, s+3-d_{1+k_B}^B\}, & \text{if } s \le d_{1+k_B}^B - 1; \\ G(2,2,s,k_A,k_B-1) + 3. \end{cases}$$

Thus, $G(2,1,0,0,0) = 3$; $G(2,1,1,0,0) = 5$;

$G(2,1,s,0,1) = 3$ for $s = 0, 1$; $G(2,1,2,0,0) = 4$;

$G(2,1,s,0,2) = 8$ for $s = 0, 1, 2$;

$G(2,1,0,1,0) = 5$; $G(2,1,1,1,0) = 6$; $G(2,1,2,1,0) = 8$;

$G(2,1,s,1,1) = 7$ for $s = 0, 1, 2$;

$G(2,1,s,1,2) = 12$ for $s = 0, 1, 2$.

• When $j_A = 1$, $j_B = 3$, $s = 0, 1, \ldots, 8$ (due to $P_2^B = 8$), $k_A = 0, 1, 2$, $k_B = 0$, we have

$$G(1,3,s,k_A,0) = \min \begin{cases} G(2,3,s+2,k_A,0) + \max\{0, s+2-d_{1+k_A}^A\}, & \text{if } s \le d_{1+k_A}^A - 1; \\ G(2,3,s,k_A-1,0) + 2. \end{cases}$$

Thus, $G(1,3,s,0,0) = 0$ for $s = 0, 1$; $G(1,3,2,0,0) = 1$;

$G(1,3,s,1,0) = 2$ for $s = 0, 1, \ldots, 6$; $G(1,3,s,1,0) = s-4$ for $s = 7, 8$;

$G(1,3,s,2,0) = 6$ for $s = 0, 1, \ldots, 8$;

• When $j_A = 1$, $j_B = 2$, $s = 0, 1, 2, 3$ (due to $P_1^B = 3$), $k_A = 0, 1, 2$, $k_B = 0, 1$, we have

$$G(1,2,s,k_A,k_B) = \min \begin{cases} G(2,2,s+2,k_A,k_B) + \max\{0, s+2-d_{1+k_A}^A\}, & \text{if } s \le d_{1+k_A}^A - 1; \\ G(2,2,s,k_A-1,k_B) + 2; \\ G(1,3,s+5,k_A,k_B) + \max\{0, s+5-d_{2+k_B}^B\}, & \text{if } s \le d_{2+k_B}^B - 1; \\ G(1,3,s,k_A,k_B-1) + 5. \end{cases}$$

Thus, $G(1,2,0,0,0) = 1$; $G(1,2,1,0,0) = 3$; $G(1,2,2,0,0) = 6$;

$G(1,2,s,0,1) = 5$ for $s = 0, 1$; $G(1,2,2,0,1) = 6$;

$G(1,2,s,1,0) = 2$ for $s = 0, 1$; $G(1,2,s,1,0) = 2s-1$ for $s = 2, 3$;

$G(1,2,s,1,1) = 7$ for $s = 0, 1, 2, 3$;

$G(1,2,s,2,0) = 6$ for $s = 0, 1, 2$; $G(1,2,3,2,0) = 7$;

$G(1,2,s,2,1) = 11$ for $s = 0, 1, 2, 3$.

• When $j_A = 1$, $j_B = 1$, $s = 0$, $k_A = 0, 1, 2$, $k_B = 0, 1, 2$, we have

$$G(1,1,0,k_A,k_B) = \min \begin{cases} G(2,1,2,k_A,k_B) + \max\{0, 2-d_{1+k_A}^A\}, & \text{if } s \le d_{1+k_A}^A - 1; \\ G(2,1,0,k_A-1,k_B) + 2; \\ G(1,2,3,k_A,k_B) + \max\{0, 3-d_{1+k_B}^B\}, & \text{if } s \le d_{2+k_B}^B - 1; \\ G(1,2,0,k_A,k_B-1) + 3. \end{cases}$$

Thus, $G(1,1,0,0,1) = 4$; $G(1,1,0,0,2) = 8$; $G(1,1,0,1,0) = 5$; $G(1,1,0,1,1) = 5$;

$G(1,1,0,1,2) = 10$; $G(1,1,0,2,0) = 7$; $G(1,1,0,2,1) = 9$; $G(1,1,0,2,2) = 14$.

The optimal solution value is $G^* = G(1,1,0,0,1) = \min\{G(1,1,0,k_A,k_B) : k_A = 0, 1, 2, k_B = 0, 1, 2\} = 4$.

The solution ϕ corresponding to state $(1,1,0,0,1)$ achieves the optimal solution value, where the job processing sequence is $J_1^A \to J_2^B \to J_2^A \to J_1^B$, the due date assignment strategy is given by $D_1^A(\phi) = 3$, $D_2^A(\phi) = 10$, $D_1^B(\phi) = 4$ and $D_2^B(\phi) = 7$. Then jobs J_1^A, J_2^A and J_2^B are effective with $Y_1^A(\phi) = 0$, $Y_2^A(\phi) = 1$ and $Y_2^B(\phi) = 0$, and job J_1^B is fully late with $Y_1^B(\phi) = 3$.

Appendix B

The implementation of Algorithm $\mathbf{A_3}$ for Example 4.1 is illustrated. Note that $n_A = 2$, $n_B = 3$, $P_1^A = 5$ and $P_2^A = 11$.

Step 1. The A-jobs and B-jobs are already numbered in the desired order, respectively.

Step 2. To find the optimal solution value, all possible u_A and u_B values have to be enumerated. As the computation details are very similar for each pair (u_A, u_B), we take $u_A = 1$ and $u_B = 1$ as a detailed illustration.

In the case where $(u_A, u_B) = (1,1)$, $v_A = 1$ and $v_B = 2$, do:

• Set $D_1^A = d_2^A = 11$, $D_2^A = d_1^A = 6$, $D_{[1]}^B = d_2^B = 7$, $D_{[2]}^B = d_3^B = 12$, $D_3^B = d_1^B = 3$, $\Delta := P_1^A + 2p^B = 13$, $\rho :=$ (), $l = 3$, $g := 1$, $h := 2$.

• In the first iteration, due to the facts that $[g = 1, D_1^A = 11 < 13 = \Delta]$ and $[h = 2 \ge 1, \Delta - p^B = 9 < 12 = D_{[2]}^B]$, set $Y_{[2]}^B = \max\{0, \Delta - D_{[2]}^B\} = 1$, $\Delta := \Delta - p^B = 9$, $\rho := (J_{[2]}^B)$, $l := 2$, $h := 1$.

• In the second iteration, due to the fact that $[g = 1, D_1^A = 11 > 9 = \Delta]$, set $\Delta := \Delta - p_1^A = 4$, $\rho := (J_1^A, J_{[2]}^B)$, $l := 1$, $g := 0$.

• In the third iteration, due to the facts that $[g = 0]$ and $[h = 1, \Delta - p^B = 0 < 7 = D_{[1]}^B]$, set $Y_{[1]}^B = \max\{0, \Delta - D_{[1]}^B\} = 0$, $\Delta := \Delta - p^B = 0$, $\rho := (J_{[1]}^B, J_1^A, J_{[2]}^B)$, $l := 0$, $h := 0$.

• Clearly, $Y_{[1]}^B < Y_{[2]}^B$. Then we have $H(1,1) = w_2^A + \lambda(w_3^B p^B + w_1^B Y_{[1]}^B + w_2^B Y_{[2]}^B) = 5 + 2 \times 4 + 0 + 3 = 16$, $J_1^B = J_{[1]}^B$, $J_2^B = J_{[2]}^B$, $D_j^B = D_{[1]}^B = 7$, $D_2^B = D_{[2]}^B = 12$.

In the same way, the other $H(u_A, u_B)$ values are $H(0,0) = H(0,1) = h(1,0) = +\infty$, $H(0,2) = 26$, $H(0,3) = 36$, $H(1,2) = 25$, $H(1,3) = 41$, $H(2,0) = 20$, $H(2,1) = 23$, $H(2,2) = 35$, $H(2,3) = 51$.

Step 3. The optimum solution value is $H^* = H(1,1) = \min\{H(u_A, u_B) : u_A = 0, 1, 2; u_B = 0, 1, 2, 3\} = 16$.

Data availability

No data was used for the research described in the article.

References

Agnetis, A., Aloulou, M.A., Kovalyov, M.Y., 2017. Integrated production scheduling and batch delivery with fixed departure times and inventory holding costs. Int. J. Prod. Res. 55 (20), 6193–6206.

Agnetis, A., Billaut, J.C., Gawiejnowicz, S., Pacciarelli, D., Soukhal, A., 2014. Multiagent Scheduling: Models and Algorithms. Springer Berlin, Heidelberg.

Agnetis, A., Mirchandani, P.B., Pacciarelli, D., Pacifici, A., 2004. Scheduling problems with two competing agents. Oper. Res. 52 (2), 229–242.

Atsmony, M., Mor, B., Mosheiov, G., 2023. Minimizing tardiness scheduling measures with generalized due-dates and a maintenance activity. Comput. Oper. Res. 152, 106133.

Baker, K.R., Smith, J.C., 2003. A multiple-criterion model for machine scheduling. J. Sched. 6 (1), 7–16.

Blazewicz, J., 1984. Scheduling preemptible tasks on parallel processors with information loss. Tech. Sci. Inform. 36, 255–258.

Blazewicz, J., Ecker, K.H., Pesch, E., Schmidt, G., Sterna, M., Weglarz, J., 2019. Handbook on Scheduling, second ed. Springer, Cham.

Blazewicz, J., Pesch, E., Sterna, M., Werner, F., 2004. Open shop scheduling problems with late work criteria. Discrete Appl. Math. 134 (1–3), 1–24.

Chen, R.B., Dong, X.Y., Yuan, J.J., Ng, C.T., Cheng, T.C.E., 2025. Single-machine preemptive scheduling with assignable due dates or assignable weights to minimize total weighted late work. European J. Oper. Res. 322 (2), 467–479.

Chen, R.B., Gao, Y., Geng, Z.C., Yuan, J.J., 2023. Revisit the scheduling problem with assignable or generalized due dates to minimize total weighted late work. Int. J. Prod. Res. 61 (22), 7630–7648.

Chen, R.B., Geng, Z.C., Lu, L.F., Yuan, J.J., Zhang, Y., 2022a. Pareto-scheduling of two competing agents with their own equal processing times. European J. Oper. Res. 301 (2), 414–431.

Chen, X., Kovalev, S., Liu, Y., Sterna, M., Chalamon, I., Blazewicz, J., 2021. Semi-online scheduling on two identical machines with a common due date to maximize total early work. Discrete Appl. Math. 290, 71–78.

Chen, R.X., Li, S.S., Feng, Q., 2024. Minimizing total weighted late work in a proportionate flow shop with job rejection. Asia-Pac. J. Oper. Res. 41 (3), 2350023.

Chen, X., Liang, Y., Sterna, M., Wang, W., Blazewicz, J., 2020. Fully polynomial time approximation scheme to maximize early work on parallel machines with common due date. European J. Oper. Res. 248 (1), 67–74.

Chen, X., Shen, X., Kovalyov, M.Y., Sterna, M., Blazewicz, J., 2022b. Alternative algorithms for identical machines scheduling to maximize total early work with a common due date. Comput. Ind. Eng. 171, 108386.

Choi, B.C., Park, M.J., Kim, K.M., Min, Y., 2021. A parallel machine scheduling problem maximizing total weighted early work. Asia-Pac. J. Oper. Res. 38 (6), 2150007.

Dereniowski, D., Kubiak, W., 2018. Shared processor scheduling. J. Sched. 21 (6), 583–593.

Fan, G.Q., Wang, J.Q., Liu, Z., 2023. Two-agent scheduling on mixed batch machines to minimize the total weighted makespan. Int. J. Prod. Res. 61 (1), 238–257.

Fomin, A., Goldengorin, B., 2022. An exact algorithm for the preemptive single machine scheduling of equal-length jobs. Comput. Oper. Res. 142, 105742.

Freud, D., Mosheiov, G., 2021. Scheduling with competing agents, total late work and job rejection. Comput. Oper. Res. 133, 105329.

Gao, Y., Yuan, J.J., 2015. Unary NP-hardness of minimizing the total deviation with generalized or assignable due dates. Discrete Appl. Math. 189, 49–52.

Garey, M.R., Johnson, D.S., 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman and Company, San Francisco, CA.

Geng, Z.C., Yuan, J.J., 2023. Single-machine scheduling of multiple projects with controllable processing times. European J. Oper. Res. 308 (3), 1074–1090.

Gordon, V.S., Proth, J.M., Chu, C., 2002. Due date assignment and scheduling: SLK, TWK and other due date assignment models. Prod. Plan. Control 13 (2), 117–132.

Gu, M., Liu, P., Lu, X., 2024. Optimal algorithms for preemptive two-agent scheduling on uniform parallel machines. Theoret. Comput. Sci. 1012, 114729.

Hall, N.G., 1986. Scheduling problems with generalized due dates. IIE Trans. 18 (2), 220–222.

Hall, N.G., Posner, M.E., 2001. Generating experimental data for computational testing with machine scheduling applications. Oper. Res. 49 (7), 854–865.

Hariri, A.M.A., Potts, C.N., Van Wassenhove, L.N., 1995. Single machine scheduling to minimize total weighted late work. ORSA J. Comput. 7 (2), 232–242.

He, C., Wu, J., Lin, H., 2022. Two-agent bounded parallel-batching scheduling for minimizing maximum cost and makespan. Discrete Optim. 45, 100698.

Jiang, Y., Wu, M., Chen, X., Dong, J., Cheng, T.C.E., Blazewicz, J., Ji, M., 2024. Online early work scheduling on parallel machines. European J. Oper. Res. 315 (3), 855–862.

Justkowiak, J.E., Kovalev, S., Kovalyov, M.Y., Pesch, E., 2023. Single machine scheduling with assignable due dates to minimize maximum and total late work. European J. Oper. Res. 308 (1), 76–83.

Kovalyov, M.Y., Oulamara, A., Soukhal, A., 2015. Two-agent scheduling with agent specific batches on an unbounded serial batching machine. J. Sched. 18 (4), 423–434.

Kuhn, H.W., 1955. The Hungarian method for the assignment problem. Naval Res. Logist. 2 (1–2), 83–97.

Leung, J.Y.T., Pinedo, M., Wan, G., 2010. Competitive two-agent scheduling and its applications. Oper. Res. 58 (2), 458–469.

Li, W.D., 2024. Improved approximation schemes for early work scheduling on identical parallel machines with a common due date. J. Oper. Res. Soc. China 12 (2), 341–350.

Li, S.S., Chen, R.X., 2022. Minimizing total weighted late work on a single-machine with non-availability intervals. J. Comb. Optim. 44 (2), 1330–1355.

Li, S.S., Chen, R.X., 2023. Competitive two-agent scheduling with release dates and preemption on a single machine. J. Sched. 26 (3), 227–249.

Li, X.Y., Liu, C.G., 2024. Two-agent scheduling on a single machine with release dates. Comput. Oper. Res. 170, 106777.

Li, M.H., Lv, D.Y., Lv, Z.G., Zhang, L.H., Wang, J.B., 2024. A two-agent resource allocation scheduling problem with slack due-date assignment and general deterioration function. Comput. Appl. Math. 43 (4), 229.

Li, S.S., Yuan, J.J., 2020. Single-machine scheduling with multi-agents to minimize total weighted late work. J. Sched. 23 (4), 497–512.

Liu, P., Gu, M., Lu, X., 2024. Two-agent scheduling in a two-machine open shop. Ann. Oper. Res. 333 (1), 275–301.

Mosheiov, G., Oron, D., Shabtay, D., 2021. Minimizing total late work on a single machine with generalized due-dates. European J. Oper. Res. 293 (3), 837–846.

Oron, D., 2021. Two-agent scheduling problems under rejection budget constraints. Omega 102, 102313.

Oron, D., Shabtay, D., Steiner, G., 2015. Single machine scheduling with two competing agents and equal job processing times. European J. Oper. Res. 244 (1), 86–99.

Pal, M., lal Mittal, M., Soni, G., Chouhan, S.S., 2023. A multi-agent system for integrated scheduling and maintenance planning of the flexible job shop. Comput. Oper. Res. 159, 106365.

Perez-Gonzalez, P., Framinan, J.M., 2014. A common framework and taxonomy for multicriteria scheduling problems with interfering and competing jobs: Multi-agent scheduling problems. European J. Oper. Res. 235 (1), 1–16.

Phosavanh, J., Oron, D., 2024. Two-agent single-machine scheduling with a rate-modifying activity. European J. Oper. Res. 312 (3), 866–876.

Phosavanh, J., Oron, D., 2025. Minimizing the number of late jobs and total late work with step-learning. European J. Oper. Res. 321 (3), 734–749.

Potts, C.N., Van Wassenhove, L.N., 1992. Single machine scheduling to minimize total late work. Oper. Res. 40 (3), 586–595.

Prata, B.A., de Abreu, L.R., Fernandez-Viagas, V., 2025. A systematic review of permutation flow shop scheduling with due-date-related objectives. Comput. Oper. Res. 177, 106989.

Qi, X., Yu, G., Bard, J.F., 2002. Single machine scheduling with assignable due dates. Discrete Appl. Math. 122 (1–3), 211–233.

Rodriguez-Ballesteros, S., Alcaraz, J., Anton-Sanchez, L., 2024. Metaheuristics for the bi-objective resource-constrained project scheduling problem with time-dependent resource costs: An experimental comparison. Comput. Oper. Res. 163, 106489.

Rolim, G.A., Nagano, M.S., 2020. Structural properties and algorithms for earliness and tardiness scheduling against common due dates and windows: A review. Comput. Ind. Eng. 149, 106803.

Sang, Y.W., Wang, J.Q., Sterna, M., Blazewicz, J., 2023. Single machine scheduling with due date assignment to minimize the total weighted lead time penalty and late work. Omega 121, 102923.

Sang, Y.W., Wang, J.Q., Sterna, M., Blazewicz, J., 2025. Single machine scheduling with the total weighted late work and rejection cost. Naval Res. Logist. 72 (2), 260–274.

Shabtay, D., 2023. A new perspective on single-machine scheduling problems with late work related criteria. Ann. Oper. Res. 322 (2), 947–966.

Shabtay, D., Mosheiov, G., Oron, D., 2022. Single machine scheduling with common assignable due date/due window to minimize total weighted early and late work. European J. Oper. Res. 303 (1), 66–77.

Sterna, M., 2007. Late work minimization in a small flexible manufacturing system. Comput. Ind. Eng. 52 (2), 210–228.

Sterna, M., 2011. A survey of scheduling problems with late work criteria. Omega 39 (2), 120–129.

Sterna, M., 2021. Late and early work scheduling: A survey. Omega 104, 102453.

Sterna, M., Czerniachowska, K., 2017. Polynomial time approximation scheme for two parallel machines scheduling with a common due date to maximize early work. J. Optim. Theory Appl. 174, 927–944.

Wang, D.J., Kang, C.C., Shiau, Y.R., Wu, C.C., Hsu, P.H., 2017. A two-agent single-machine scheduling problem with late work criteria. Soft Comput. 21 (8), 2015–2033.

Wang, X., Ren, T., Bai, D., Chu, F., Yu, Y., Meng, F., Wu, C.C., 2024. Scheduling a multi-agent flow shop with two scenarios and release dates. Int. J. Prod. Res. 62 (1–2), 421–443.

Wang, D.J., Yin, Y., Xu, J., Wu, W.H., Cheng, S.R., Wu, C.C., 2015. Some due date determination scheduling problems with two agents on a single machine. Int. J. Prod. Econ. 168, 81–90.

Xu, L., Mak, S., Minaricova, M., Brintrup, A., 2024. On implementing autonomous supply chains: A multi-agent system approach. Comput. Ind. 161, 104120.

Yin, Y., Cheng, S., Cheng, T.C.E., Wu, C.C., Wu, W.H., 2012. Two-agent single-machine scheduling with assignable due dates. Appl. Math. Comput. 219 (4), 1674–1685.
Yin, Y., Wang, D.J., Cheng, T.C.E., 2020. Due Date-Related Scheduling with Two Agents: Models and Algorithms. Springer, Singapore.
Yin, Y., Xu, J., Cheng, T.C.E., Wu, C.C., Wang, D.J., 2016. Approximation schemes for single-machine scheduling with a fixed maintenance activity to minimize the total amount of late work. Naval Res. Logist. 213 (2), 172–183.
Zhang, L.H., Lv, D.Y., Wang, J.B., 2023. Two-agent slack due-date assignment scheduling with resource allocations and deteriorating jobs. Mathematics 11 (12), 2737.
Zhang, Y., Yuan, J.J., 2019. A note on a two-agent scheduling problem related to the total weighted late work. J. Comb. Optim. 37 (3), 989–999.
Zhang, Y., Yuan, J.J., Ng, C.T., Cheng, T.C.E., 2021. Pareto-optimization of three-agent scheduling to minimize the total weighted completion time, weighted number of tardy jobs, and total weighted late work. Naval Res. Logist. 68 (3), 378–393.