Pareto-optimization of three-agent scheduling to minimize the total weighted completion time, weighted number of tardy jobs, and total weighted late work

Yuan Zhang[1], Jinjiang Yuan[1*], C.T. Ng[2], T.C.E. Cheng[2]

[1]School of Mathematics and Statistics, Zhengzhou University,

Zhengzhou, Henan 450001, People's Republic of China

[2]Logistics Research Centre, Department of Logistics and Maritime Studies,

The Hong Kong Polytechnic University, Hong Kong SAR, People's Republic of China

---

Abstract We consider three-agent scheduling on a single machine in which the criteria of the three agents are to minimize the total weighted completion time, the weighted number of tardy jobs, and the total weighted late work, respectively. The problem is to find the set of all the Pareto-optimal points, i.e., the Pareto frontier, and their corresponding Pareto-optimal schedules. Since the above problem is unary NP-hard, we study the problem under the restriction that the jobs of the first agent have inversely agreeable processing times and weights, i.e., the smaller the processing time of a job is, the greater its weight is. For this restricted problem, which is NP-hard, we present a pseudo-polynomial-time algorithm to find the Pareto frontier. We also show that, for various special versions, the time complexity of solving the problem can be further reduced.

Keywords: scheduling; three-agent Pareto-optimization; total weighted completion time; weighted number of tardy jobs; total weighted late work

Email address:
15136173837@163.com (Yuan Zhang),
yuanjj@zzu.edu.cn (Jinjiang Yuan),
daniel.ng@polyu.edu.hk (C.T. Ng),
edwin.cheng@polyu.edu.hk (T.C.E. Cheng).

*Corresponding author. Email address: yuanjj@zzu.edu.cn

1 Introduction

The total weighted completion time and the weighted number of tardy jobs are two basic and important criteria in scheduling research. They have a wide range of applications in machinery manufacturing, light industry, agriculture, logistics, service industry, and other industries. The goal of minimizing the total weighted completion time reflects the decision maker's desire to save the time spent on each object. For example, the service industry values customer satisfaction, so reducing the sum of waiting times for all customers is a major goal for managers, where the more important customers are given more weights. On the other hand, the weighted number of tardy jobs is an objective function related to the task due dates, which reflects the decision maker's desire to complete each task on or before its due date. When a task is not finished on time, the processing of the task is a failure. For example, in steel production, hot forging needs to take place before the metal billet cools off. Once the hot forging for a metal billet is not finished before it is cool off, the billet is useless. The factory naturally would like to produce as many useful forgings as possible. In the single-machine setting, Smith (1956) first studied problem $1||\sum w_jC_j$, while Lawler and Moore (1969), Moore (1968), and Sahni (1976) first considered problem $1||\sum w_jU_j$. On the other hand, Blazewicz (1984) introduced late work as a scheduling criterion, which is meaningful in the contexts of computerized control systems, processing of perishable goods in agricultural production, etc. The goal of minimizing the total weighted late work reflects the decision maker's desire to reduce the negative impacts of the processing durations beyond their corresponding due dates, given the fact that even if a job is tardy, the part that is processed before the due date is still valuable. For example, the harvesting and planting of crops depends on climate and time. Given the variety of crops, each field must be planted or harvested at a specific time, after which agricultural activities become meaningless. Farmers would like to maximize their profits by rationalizing their farming activities or, equivalently, minimizing the unproductive agricultural activities. We can model this goal as minimizing the total weighted late work whereby we treat each piece of land as a job and assign different weights to different lands that reflect the values of the crops.

Extending classical scheduling research that considers a single agent, Baker and Smith (2003), and Agnetis et al. (2004) pioneered two-agent scheduling research. Since then, two-agent (and later multi-agent) scheduling has become a popular stream of scheduling research. However, in the literature, no research on multi-agent scheduling has considered the above three criteria together. In fact, it is of practical significance to combine the above three objective functions. For example, consider the following three types of parts processing in a flexible manufacturing system (FMS). The first type of parts (jobs) has no special requirement for the processing environment. In order to save the time cost spent on each job, and considering the different importance of different jobs, the system would like to minimize the total weighted completion time of the first type of jobs. The second type of parts (jobs) needs to be repeatedly polished for a certain time (which can be regarded as jobs' processing times), but polishing after the parts have cooled down will not produce the desired effect. So the longer the polishing time before the parts cool down is, the higher the precision and the better the quality of the parts are. The time point at which a part is completely cooled can be regarded as the due date of the part. For this type of jobs, it is desirable that the processing duration after the due date be as

short as possible. Given the different importance of different jobs, we can model this requirement as minimizing the total weighted late work. The third type of parts (jobs) needs to be forged before they cool (due date) and the system sets a fixed forging time (processing time) for each part. Unlike the second type of jobs, this type of jobs is useless once it has not been completed before it cools down. So the system would like as many jobs as possible to be completed before the due dates. Taking the weights of the jobs into account, we can model this requirement as minimizing the weighted number of tardy jobs. Because the machine in the FMS has the ability to process different kinds of parts as the products change, it is possible to freely switch the processing of the three kinds of parts, provided that each part cannot be interrupted once the processing begins. We assume that the tool change time is negligible relative to the machining times of the parts. The above three types of jobs can be regarded as belonging to three different agents. In order to achieve a proper balance in the production of the three types of jobs in the system, it is reasonable to consider seeking the Pareto-optimal schedule for them.

In this paper we consider Pareto-optimization of three-agent scheduling in which the criteria of the three agents are to minimize the total weighted completion time, weighted number of tardy jobs, and total weighted late work, respectively. We focus on the case where the jobs of the first agent have inversely agreeable processing times and weights, i.e., the smaller the processing time of a job is, the greater its weight is. This reflects a realistic phenomenon in practice: The market demand for a current product with reliable and good performance is high, so its weight is large, and its processing time is relatively short due to the use of mature processing technology to produce it. On the other hand, a newly developed product, with a long processing time due to the use of immature technology for its production, is not easily accepted by the consumers, so its market demand is low and its weight is small. Another case where the jobs have inversely agreeable due dates and weights is worthy considering. It reflects the phenomenon that the greater the urgency of a task is, the smaller its due date is. Furthermore, the above situations include the special cases where the jobs have the same processing time, the same weight, or the same due date, which are common in real-world production.

1.1 Problem formulation

Let $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$ be a set of jobs, all available at time 0, to be non-preemptively processed on a single machine. Let A_1, A_2, and A_3 be three competing agents. Each job in $\mathcal{J}$ belongs to exactly one of the agents A_1, A_2, and A_3. For each $z \in \{1, 2, 3\}$, we use $\mathcal{J}^{(z)}$ to denote the set of jobs of agent A_z (called the A_z-jobs) in $\mathcal{J}$ and define $\mathcal{J}^{(z)} = \{J_1^{(z)}, J_2^{(z)}, \ldots, J_{n_z}^{(z)}\}$. Then $n_1 + n_2 + n_3 = n$ and $(\mathcal{J}^{(1)}, \mathcal{J}^{(2)}, \mathcal{J}^{(3)})$ forms a partition of $\mathcal{J}$. For each job $J_j^{(z)} \in \mathcal{J}^{(z)}$, where $z \in \{1, 2, 3\}$ and $1 \leq j \leq n_z$, we have the following related parameters.

- $p_j^{(z)}$ is the processing time of job $J_j^{(z)}$.
- $w_j^{(z)}$ is the weight of job $J_j^{(z)}$.
- $d_j^{(z)}$ is the due date of job $J_j^{(z)}$.

We assume that all the parameters $p_j^{(z)}, w_j^{(z)}$, and $d_j^{(z)}$ are positive integers. When a job $J_j \in \mathcal{J}$ is not specified, we use p_j, w_j, and d_j to denote the processing time, the weight, and the due date

of J_j, respectively. The problems studied in this paper allow us to consider only the schedules in which the jobs in $\mathcal{J}$ are processed consecutively from time 0 without idle times between the jobs, and each of such schedules is called a *tight schedule*. For a given schedule σ of $\mathcal{J}$, we use $J_{\sigma(i)}$ to denote the i-th completed job in σ, where $i \in \{1, 2, \ldots, n\}$. Since σ is tight, we use the permutation $\sigma = (J_{\sigma(1)}, J_{\sigma(2)}, \ldots, J_{\sigma(n)})$ to denote the schedule σ. For each job $J_j^{(z)} \in \mathcal{J}$, we use $S_j^{(z)}(\sigma)$ and $C_j^{(z)}(\sigma)$ to denote the starting time and completion time of job $J_j^{(z)}$ in σ, respectively. Then we have $C_j^{(z)}(\sigma) = S_j^{(z)}(\sigma) + p_j^{(z)}$.

The *late work* of job $J_j^{(z)}$ in a schedule σ, denoted by $Y_j^{(z)}(\sigma)$, is the amount of processing of job $J_j^{(z)}$ that is scheduled after its due date $d_j^{(z)}$ in σ. Since preemption is not allowed, we have

$$Y_j^{(z)}(\sigma) = \begin{cases} 0, & \text{if } C_j^{(z)}(\sigma) \le d_j^{(z)}, \\ C_j^{(z)}(\sigma) - d_j^{(z)}, & \text{if } d_j^{(z)} < C_j^{(z)}(\sigma) < d_j^{(z)} + p_j^{(z)}, \\ p_j^{(z)}, & \text{if } C_j^{(z)}(\sigma) \ge d_j^{(z)} + p_j^{(z)}, \end{cases}$$

or, equivalently,

$$Y_j^{(z)}(\sigma) = \min\{\max\{C_j^{(z)}(\sigma) - d_j^{(z)}, 0\}, p_j^{(z)}\}.$$

We use the following terms and notation throughout the paper.

- $J_j^{(z)}$ is called *early* in σ if $Y_j^{(z)}(\sigma) = 0$ or, equivalently, $C_j^{(z)}(\sigma) \le d_j^{(z)}$. In this case, we set $U_j^{(z)}(\sigma) = 0$.
- $J_j^{(z)}$ is called *strictly early* in σ if $C_j^{(z)}(\sigma) < d_j^{(z)}$.
- $J_j^{(z)}$ is called *tardy* in σ if $Y_j^{(z)}(\sigma) > 0$ or, equivalently, $C_j^{(z)}(\sigma) > d_j^{(z)}$. In this case, we set $U_j^{(z)}(\sigma) = 1$.
- $J_j^{(z)}$ is called *partially early* in σ if $0 < Y_j^{(z)}(\sigma) < p_j^{(z)}$ or, equivalently, $C_j^{(z)}(\sigma) - p_j^{(z)} < d_j^{(z)} < C_j^{(z)}(\sigma)$.
- $J_j^{(z)}$ is called *late* in σ if $Y_j^{(z)}(\sigma) = p_j^{(z)}$ or, equivalently, $S_j^{(z)}(\sigma) \ge d_j^{(z)}$.
- $J_j^{(z)}$ is called *non-late* in σ if $Y_j^{(z)}(\sigma) < p_j^{(z)}$ or, equivalently, $S_j^{(z)}(\sigma) < d_j^{(z)}$. Clearly, a job is non-late if and only if it is either an early job or a partially early job.
- $\gamma^{(1)}(\sigma) = \sum_{j=1}^{n_1} w_j^{(1)} C_j^{(1)}(\sigma)$ is the total weighted completion time of the A_1-jobs in σ.
- $\gamma^{(2)}(\sigma) = \sum_{j=1}^{n_2} w_j^{(2)} U_j^{(2)}(\sigma)$ is the weighted number of tardy A_2-jobs in σ.
- $\gamma^{(3)}(\sigma) = \sum_{j=1}^{n_3} w_j^{(3)} Y_j^{(3)}(\sigma)$ is the total weighted late work of the A_3-jobs in σ.

Definition 1.1. *Consider two m-vectors $\boldsymbol{u} = (u_1, u_2, \ldots, u_m)$ and $\boldsymbol{v} = (v_1, v_2, \ldots, v_m)$.*

*(i) We say that **$\boldsymbol{u}$ dominates $\boldsymbol{v}$**, denoted by $\boldsymbol{u} \preceq \boldsymbol{v}$, if $u_i \le v_i$ for $i = 1, 2, \ldots, m$.*

*(ii) We say that **$\boldsymbol{u}$ strictly dominates $\boldsymbol{v}$**, denoted by $\boldsymbol{u} \prec \boldsymbol{v}$, if $\boldsymbol{u} \preceq \boldsymbol{v}$ and $\boldsymbol{u} \ne \boldsymbol{v}$.*

Definition 1.2. *Suppose that agent A_z wants to minimize its criterion $\gamma^{(z)}$, where $z \in \{1, 2, 3\}$. For each schedule σ for $\mathcal{J}$, we call $(\gamma^{(1)}(\sigma), \gamma^{(2)}(\sigma), \gamma^{(3)}(\sigma))$ an **objective vector** of the three agents (A_1, A_2, A_3). A schedule σ for $\mathcal{J}$ is called **Pareto-optimal** if there is no other schedule π for $\mathcal{J}$ such that $(\gamma^{(1)}(\pi), \gamma^{(2)}(\pi), \gamma^{(3)}(\pi))$ strictly dominates $(\gamma^{(1)}(\sigma), \gamma^{(2)}(\sigma), \gamma^{(3)}(\sigma))$. In this case, we also say that $(\gamma^{(1)}(\sigma), \gamma^{(2)}(\sigma), \gamma^{(3)}(\sigma))$ is the **Pareto-optimal point** corresponding to schedule σ. We define the **Pareto frontier** as the set of all the Pareto-optimal points.*

In this paper we focus on Pareto-optimization, i.e., finding the set of all the Pareto-optimal points, of the scheduling problem

$$1|\beta|(\gamma^{(1)},\gamma^{(2)},\gamma^{(3)}) \tag{1}$$

and its subproblems $1|\beta|(\gamma',\gamma'')$ with $\gamma',\gamma'' \in \{\gamma^{(1)},\gamma^{(2)},\gamma^{(3)}\}$, where β is a set of conditions for the job instance $\mathcal{J}=\mathcal{J}^{(1)}\cup\mathcal{J}^{(2)}\cup\mathcal{J}^{(3)}$. In particular, we use the following two conditions in the β-field.

• $p_j^{(1)} \uparrow\downarrow w_j^{(1)}$, which means that the processing times and the weights of the A_1-jobs are inversely agreeable, i.e., $p_i^{(1)} < p_j^{(1)}$ implies $w_i^{(1)} \geq w_j^{(1)}$.

• $d_j^{(3)} \uparrow\downarrow w_j^{(3)}$, which means that the due dates and the weights of the A_3-jobs are inversely agreeable, i.e., $d_i^{(3)} < d_j^{(3)}$ implies $w_i^{(3)} \geq w_j^{(3)}$.

The aim of the problem in (1) is to generate the Pareto frontier and, for each Pareto-optimal point in the Pareto frontier, the corresponding Pareto-optimal schedule.

1.2 Complexity description

By using a reduction from the unary NP-complete 3-Partition Problem (Garey and Johnson, 1979), Lawler (1977) showed that problem $1||\sum w_jT_j$ is unary NP-hard. In Lawler's proof, the constructed job instance consists of two families A and B. The A-jobs have a common due date 0 and the B-jobs have a common processing time $p^{(B)}$. Moreover, each B-job has a very large weight so that in every feasible schedule all the B-jobs must be early. Since the total weighted tardiness of the A-jobs is clearly equal to the total weighted completion time of the A-jobs, Lawler's proof can also be used to show that both problems $1|p_j^{(B)}=p^{(B)}|\sum w_j^{(A)}C_j^{(A)} : \sum U_j^{(B)} \leq 0$ and $1|p_j^{(B)}=p^{(B)}|\sum w_j^{(A)}C_j^{(A)} : \sum Y_j^{(B)} \leq 0$ are unary NP-hard. Then the problems $1|p_j^{(2)}=p^{(2)}|(\sum w_j^{(1)}C_j^{(1)},\sum U_j^{(2)})$ and $1|p_j^{(3)}=p^{(3)}|(\sum w_j^{(1)}C_j^{(1)},\sum Y_j^{(3)})$ are unary NP-hard, which further implies that problem $1||(\gamma^{(1)},\gamma^{(2)},\gamma^{(3)})$ is unary NP-hard.

Potts and Van Wassenhove (1992) showed that the single-machine problem to minimize the total late work is NP-hard. Then problem $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)},\gamma^{(2)},\gamma^{(3)})$ is also NP-hard, so as the following problems: $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}, d_j^{(3)} \uparrow\downarrow w_j^{(3)}|(\gamma^{(1)},\gamma^{(2)},\gamma^{(3)})$, $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)},\gamma^{(3)})$, $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}, d_j^{(3)} \uparrow\downarrow w_j^{(3)}|(\gamma^{(1)},\gamma^{(3)})$, $1|d_j^{(3)} \uparrow\downarrow w_j^{(3)}|(\gamma^{(2)},\gamma^{(3)})$, and $1||(\gamma^{(2)},\gamma^{(3)})$. Chen et al. (2019) showed that problem $1|p_j^{(A)}=p^{(A)}|\sum C_j^{(A)} : \sum U_j^{(B)} \leq Q$ is NP-hard. Then problem $1|p_j^{(1)}=p^{(1)}|(\sum C_j^{(1)},\sum U_j^{(2)})$ is NP-hard. Thus problem $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)},\gamma^{(2)})$ is NP-hard. So we attempt to find a pseudo-polynomial-time algorithm to solve them.

1.3 Our contributions

We study in this paper Pareto-optimization of the scheduling problem $1|\beta|(\gamma^{(1)},\gamma^{(2)},\gamma^{(3)})$ with special choices in the β-field and of parameters $p_j^{(z)}, w_j^{(z)}$, and $d_j^{(z)}$. We summarize our findings on the computational complexity of the variants of the problem studied in Table 1, where $Q^{(z)}$ is an arbitrary upper bound on the criterion $\gamma^{(z)}$ for $z \in \{1,2,3\}$.

Table 1: Complexity of $1|\beta|(\gamma^{(1)},\gamma^{(2)},\gamma^{(3)})$

Scheduling problem	Complexity	Reference
$1\|p_j^{(1)} \uparrow\downarrow w_j^{(1)}\|(\gamma^{(1)},\gamma^{(2)},\gamma^{(3)})$	$O(n_1n_2n_3^2Q^{(1)}Q^{(2)}Q^{(3)})$	Theorem 4.1
$1\|p_j^{(1)} \uparrow\downarrow w_j^{(1)}, d_j^{(3)} \uparrow\downarrow w_j^{(3)}\|(\gamma^{(1)},\gamma^{(2)},\gamma^{(3)})$	$O(n_1n_2n_3Q^{(1)}Q^{(2)}Q^{(3)})$	Corollary 4.1
$1\|p_j^{(1)} \uparrow\downarrow w_j^{(1)}\|(\gamma^{(1)},\gamma^{(2)})$	$O(n_1n_2Q^{(1)}Q^{(2)})$	Corollary 4.2
$1\|p_j^{(1)} \uparrow\downarrow w_j^{(1)}\|(\gamma^{(1)},\gamma^{(3)})$	$O(n_1n_3^2Q^{(1)}Q^{(3)})$	Corollary 4.3
$1\|p_j^{(1)} \uparrow\downarrow w_j^{(1)}, d_j^{(3)} \uparrow\downarrow w_j^{(3)}\|(\gamma^{(1)},\gamma^{(3)})$	$O(n_1n_3Q^{(1)}Q^{(3)})$	Corollary 4.3
$1\|\|(\gamma^{(2)},\gamma^{(3)})$	$O(n_2n_3^2Q^{(2)}Q^{(3)})$	Corollary 4.4
$1\|d_j^{(3)} \uparrow\downarrow w_j^{(3)}\|(\gamma^{(2)},\gamma^{(3)})$	$O(n_2n_3Q^{(2)}Q^{(3)})$	Corollary 4.4
$1\|p_j^{(2)}=p^{(2)}, p_j^{(3)}=p^{(3)}\|(\sum C_j^{(1)}, \sum U_j^{(2)}, \sum Y_j^{(3)})$	$O(n_1^3n_2^3n_3^3)$	Theorem 5.1
$1\|p_j^{(2)}=p^{(2)}\|(\sum C_j^{(1)}, \sum U_j^{(2)})$	$O(nn_2)$	Theorem 5.2
$1\|p_j^{(2)}=p^{(2)}, p_j^{(3)}=p^{(3)}\|(\sum U_j^{(2)}, \sum Y_j^{(3)})$	$O(nn_2n_3)$	Theorem 5.2
$1\|p_j^{(3)}=p^{(3)}\|(\sum C_j^{(1)}, \sum Y_j^{(3)})$	$O(n_1^2n_3^2)$	Corollary 5.1

1.4 Organization of the paper

We organize the rest of the paper as follows: In Section 2 we review the related known results in the literature. In Section 3 we present some preliminaries for algorithm design. In Section 4 we provide a pseudo-polynomial-time algorithm to solve problem $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)},\gamma^{(2)},\gamma^{(3)})$ and discuss various implications. In Section 5 we present a polynomial-time algorithm to solve problem $1|p_j^{(2)}=p^{(2)}, p_j^{(3)}=p^{(3)}|(\sum C_j^{(1)}, \sum U_j^{(2)}, \sum Y_j^{(3)})$ and discuss various implications. We conclude the paper and suggest future research topics in the last section.

2 Literature review

In order to be consistent with the notation in the literature, we also use A and B to denote two agents. Given that Agnetis et al. (2014) provided a detailed review of the literature on agent scheduling before 2014, we only review studies on competing-agent scheduling on a single machine involving the above three objective functions published after 2013.

Agnetis et al. (2019) applied the concept of price of fairness in two-agent scheduling where agent A wants to minimize the total completion time of the A-jobs, while agent B seeks to minimize the maximum tardiness of the B-jobs (with a common due date). Then they extended their study to consider the problem in which both agents A and B seek to minimize the total completion time of their own jobs. Chen et al. (2019) proved that problem $1|p_j^{(A)}=p^{(A)}|\sum C_j^{(A)} : \sum U_j^{(B)} \leq Q$ is NP-hard. Chen and Li (2019) presented either a polynomial-time or a pseudo-polynomial-time algorithm to solve several cumulative deterioration two-agent scheduling problems, seeking to minimize the objective value of agent A, while keeping the objective value of agent B not exceeding a given

bound. The scheduling criteria of agents A and B are combinations of the following regular criteria: the maximum cost, the total completion time, and the (weighted) number of tardy jobs. Cheng et al. (2014) studied two-agent single-machine scheduling to minimize the weighted sum of the total completion time of the jobs of agent A and the total tardiness of the jobs of agent B. They provided branch-and-bound algorithms to solve the problem. In addition, they presented a simulated annealing algorithm and two genetic algorithms to obtain near-optimal solutions. Cheng et al. (2019) presented an effective algorithm to generate the non-dominated solutions for single-machine scheduling problem with two competing agents in which the performance criteria are the mean lateness and the number of tardy jobs. Choi and Chung (2014) and Choi et al. (2019) studied two-agent single-machine scheduling to optimize the performance measure for agent A, while maintaining the weighted number of just-in-time jobs for agent B at or above a given threshold. The performance measures for agent A are the total weighted completion time and the weighted number of tardy jobs. They showed that both problems are unary NP-hard. They also studied the computational complexity of the special cases where the weights or the processing times of each agent are identical. Dover and Shabtay (2016) showed that problem $1|CO, r_j^{(i)}, p_j^{(i)} = 1, \sum w_j^{(A)} C_j^{(A)} \leq K_1, \sum w_j^{(B)} U_j^{(B)} \leq K_2|-$ is NP-complete; the high multiplicity variant of problem $1|CO, r_j^{(i)}, p_j^{(i)} = 1, \sum C_j^{(A)} \leq K_1, \sum w_j^{(B)} U_j^{(B)} \leq K_2|-$ is NP-complete; and all the variants of problem $1|CO, r_j^{(i)}, p_j^{(i)} = 1|(\sum C_j^{(A)}, \sum C_j^{(B)})$ are solvable in polynomial time. Oron et al. (2015) studied various two-agent scheduling problems on a single machine with equal job processing times. Especially, they showed that problem $1|p_j = 1|\sum w_j^{(A)} U_j^{(A)} : \sum w_j^{(B)} U_j^{(B)} \leq Q$ is binary NP-hard and problem $1|p_j = 1|(\sum w_j^{(A)} C_j^{(A)}, \sum w_j^{(B)} U_j^{(B)})$ is solvable in $O(n_A n_B^2 \min\{\overline{F_A}, \overline{F_B}\})$ time, where $\overline{F_A} = \sum_{j=n_B+1}^{n_A+n_B} j w_j^{(A)}$ and $\overline{F_B} = \sum_{j=1}^{n_B} w_j^{(B)}$. As special cases, problems $1|p_j = 1|(\sum C_j^{(A)}, \sum w_j^{(B)} U_j^{(B)})$ and $1|p_j = 1|(\sum w_j^{(A)} C_j^{(A)}, \sum U_j^{(B)})$ are solvable in $O(n_A^2 n_B^2 (n_A + n_B))$ time and $O(\max\{n_A \log n_A, n_B(n_A + n_B)\})$ time, respectively. Wan et al. (2016) provided a strongly polynomial-time algorithm for the Pareto-scheduling problem $1||(\sum U_j^{(A)}, f_{\max}^{(B)})$. Wan et al. (2020) showed that problem $1|p_j = 1|\sum w_j^{(A)} C_j^{(A)} : \sum w_j^{(B)} U_j^{(B)} \leq Q$ is binary NP-hard. Yin et al. (2016) studied the problems of minimizing the performance criterion of agent A in the form $\varphi d^{(A)} + \sum w_j^{(A)} U_j^{(A)}$, while keeping the objective value of $f_{\max}^{(B)}$ (resp., $\sum C_j^{(B)}$, $\sum w_j^{(B)} C_j^{(B)}$, and $\sum w_j^{(B)} U_j^{(B)}$) no greater than a given limit Q under the common due date and slack due dates, respectively. They proved that all of the problems are NP-hard in the ordinary sense and discussed possible solution algorithms. Zhang and Wang (2017), and Zhang and Yuan (2019) studied two-agent scheduling on a single machine to minimize the total weighted late work of agent A, subject to the restriction that the maximum cost of agent B is bounded. They showed that the problem is solvable in pseudo-polynomial time in general and in polynomial time when the A-jobs have identical processing times.

Among the studies on multi-agent scheduling, two-agent scheduling makes up the majority, while there are a few studies involving more than two agents. Lee and Wang (2014; 2017) considered three-agent single-machine scheduling. In Lee and Wang (2014), the objective is to minimize the total weighted completion time of the A-jobs, given that the maximum completion time of the B-jobs does not exceed an upper bound and the maintenance activity of agent C must be performed within a specified period of time. In Lee and Wang (2017), their aim is to minimize the makespan

of the A-jobs, given that the maximum tardiness of the B-jobs cannot exceed a given bound and that a maintenance activity of agent C must be completed within a specified maintenance window. They proposed a branch-and-bound algorithm and a genetic algorithm to obtain optimal and approximate solutions, respectively, in the two studies. Yuan (2017) showed that when m is fixed, problem $1|CO|\sum_{i=1}^{m_1}\alpha_i(\sum_{J_j\in\mathcal{J}^{(i)}}U_j)+\sum_{i=m_1+1}^{m}\alpha_i C_{\max}^{(i)}$ can be solved in polynomial time. Li and Yuan (2020) considered multi-agent single-machine scheduling where each agent wants to minimize its total weighted late work. They showed that the problem is unary NP-hard even when all the jobs have a unit processing time. When the number of agents is fixed, they provided a pseudo-polynomial dynamic programming algorithm and a $(1+\epsilon)$-approximate Pareto-optimal frontier for the problem. Recently, Yuan (2016; 2018) and Yuan et al. (2020) provided some new complexity results on multi-agent scheduling.

3 Preliminaries for algorithm design

Consider problem $1|\beta|(\gamma^{(1)},\gamma^{(2)},\gamma^{(3)})$ and let $\mathcal{J}=\mathcal{J}^{(1)}\cup\mathcal{J}^{(2)}\cup\mathcal{J}^{(3)}$ be the job instance, where $\mathcal{J}^{(z)}=\{J_1^{(z)},J_2^{(z)},\ldots,J_{n_z}^{(z)}\}$ for $z=1,2,3$. For a subset $\mathcal{F}\subseteq\mathcal{J}$, we use $p(\mathcal{F})$ to denote the total processing time of the jobs in $\mathcal{F}$. In the preprocessing procedure for solving the above problem, we re-number the A_1-jobs in non-decreasing order of their Smith ratios $p_j^{(1)}/w_j^{(1)}$ such that

$$p_1^{(1)}/w_1^{(1)}\le p_2^{(1)}/w_2^{(1)}\le\cdots\le p_{n_1}^{(1)}/w_{n_1}^{(1)}, \tag{2}$$

re-number the A_2-jobs in the earliest due date (EDD) order such that

$$d_1^{(2)}\le d_2^{(2)}\le\cdots\le d_{n_2}^{(2)}, \tag{3}$$

and re-number the A_3-jobs in the EDD order such that

$$d_1^{(3)}\le d_2^{(3)}\le\cdots\le d_{n_3}^{(3)}, \tag{4}$$

with ties being broken by the largest weight first (LW) rule, i.e.,

$$w_i^{(3)}\ge w_j^{(3)} \text{ if } i<j \text{ and } d_i^{(3)}=d_j^{(3)}. \tag{5}$$

Clearly, the preprocessing procedure can be executed in $O(n_1\log n_1+n_2\log n_2+n_3\log n_3)=O(n\log n)$ time. Throughout the rest of the paper, we use the job indices presented in (2)-(5). Moreover, for $z\in\{1,2,3\}$ and $j\in\{1,2,\ldots,n_z\}$, we define

$$\begin{cases}\mathcal{J}_j^{(z)} &= \{J_1^{(z)},J_2^{(z)},\ldots,J_j^{(z)}\},\\ P_j^{(z)} &= p_1^{(z)}+p_2^{(z)}+\cdots+p_j^{(z)}.\end{cases} \tag{6}$$

Then all the values $P_j^{(z)}=p(\mathcal{J}_j^{(z)})$, $z\in\{1,2,3\}$ and $j\in\{1,2,\ldots,n_z\}$, can be obtained in $O(n)$ time. From the above indexing rule for the jobs, we easily obtain the following lemma.

Lemma 3.1. *For a given instance $\mathcal{J}$, we have the following statements.*

(i) If the condition $p_j^{(1)} \uparrow\downarrow w_j^{(1)}$ holds, then $p_1^{(1)} \le p_2^{(1)} \le \cdots \le p_{n_1}^{(1)}$ and $w_1^{(1)} \ge w_2^{(1)} \ge \cdots \ge w_{n_1}^{(1)}$.

(ii) If the condition $d_j^{(3)} \uparrow\downarrow w_j^{(3)}$ holds, then $d_1^{(3)} \le d_2^{(3)} \le \cdots \le d_{n_3}^{(3)}$ and $w_1^{(3)} \ge w_2^{(3)} \ge \cdots \ge w_{n_3}^{(3)}$.

Note that $(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)}) = (\sum w_j^{(1)} C_j^{(1)}, \sum w_j^{(2)} U_j^{(2)}, \sum w_j^{(3)} Y_j^{(3)})$. In a schedule σ for $\mathcal{J}$, a job $J_j \in \mathcal{J}$ is called *valid* if J_j is either an A_1-job, or an early A_2-job, or a non-late A_3-job. A job that is not valid in σ is called *invalid* in σ. In a Pareto-optimal schedule for problem $1|\beta|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$, we can always schedule the invalid jobs after all the valid jobs. Thus, in our discussion and algorithm design, we only consider schedules of the valid jobs, called *valid-schedules*. Since all the schedules considered in this paper are tight, we denote a valid-schedule σ by

$$\sigma = (J_{\sigma(1)}, J_{\sigma(2)}, \ldots, J_{\sigma(n')}), \tag{7}$$

where $n_1 \le n' \le n = n_1 + n_2 + n_3$ and $\{J_{\sigma(1)}, J_{\sigma(2)}, \ldots, J_{\sigma(n')}\}$ consists of all the valid jobs in σ. This means that n' is always used to denote the number of valid jobs. Given a valid-schedule σ and a subset of valid jobs $\mathcal{F}$, we use $\vec{\mathcal{F}}_\sigma$ to denote the order of the jobs of $\mathcal{F}$ in σ. As in Hariri et al. (1995), we introduce the following definitions in our research, which are slightly different from that in Hariri et al. (1995).

Definition 3.1. *Let $\sigma = (J_{\sigma(1)}, J_{\sigma(2)}, \ldots, J_{\sigma(n')})$ be a valid-schedule for $\mathcal{J}$. For two jobs J_i and J_j in $\mathcal{J}$, we use the notation $J_i \prec_\sigma J_j$ to indicate that J_i is scheduled before J_j in σ. A non-late A_3-job $J_i^{(3)}$ is called **deferred** (from the job-index order of the A_3-jobs) in σ if there is a non-late A_3-job $J_j^{(3)}$ with $i < j$ such that $J_j^{(3)} \prec_\sigma J_i^{(3)}$. In this case, we also say that $J_j^{(3)} J_i^{(3)}$ forms a **reversed pair** in σ.*

Note that an A_3-job $J_i^{(3)}$ is non-late, i.e., valid, in a valid-schedule σ for $\mathcal{J}$ if and only if $S_i^{(3)}(\sigma) < d_i^{(3)}$. Similar to the discussion in Hariri et al. (1995), we have the following result about the deferred jobs and reversed pairs.

Lemma 3.2. *Let $\sigma = (J_{\sigma(1)}, J_{\sigma(2)}, \ldots, J_{\sigma(n')})$ be a schedule for $\mathcal{J}$. For each reversed pair $J_j^{(3)} J_i^{(3)}$ of non-late A_3-jobs, $J_j^{(3)}$ must be strictly early in σ.*

Proof. The assumption implies that $i < j$, $J_j^{(3)} \prec_\sigma J_i^{(3)}$, and $J_i^{(3)}$ is non-late in σ. Then we have $d_i^{(3)} \le d_j^{(3)}$ and $S_i^{(3)}(\sigma) < d_i^{(3)}$. From the fact that $J_j^{(3)} \prec_\sigma J_i^{(3)}$, we have $C_j^{(3)}(\sigma) \le S_i^{(3)}(\sigma) < d_i^{(3)} \le d_j^{(3)}$. Thus, $J_j^{(3)}$ is strictly early in σ. □

The following lemmas provide some useful properties of the Pareto-optimal schedules for problem $1|\beta|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$.

Lemma 3.3. *For each Pareto-optimal point (C, U, Y), there exists a corresponding Pareto-optimal schedule σ such that the following statements hold for the A_1-jobs.*

(i) For every two consecutively processed A_1-jobs $J_i^{(1)}$ and $J_j^{(1)}$ with $J_i^{(1)} \prec_\sigma J_j^{(1)}$, it holds that $i < j$, implying $p_i^{(1)}/w_i^{(1)} \le p_j^{(1)}/w_j^{(1)}$.

(ii) If $p_j^{(1)} \uparrow\downarrow w_j^{(1)}$ is a condition in the β-field, then the A_1-jobs are scheduled in their job-index order, i.e., $J_1^{(1)} \prec_\sigma J_2^{(1)} \prec_\sigma \cdots \prec_\sigma J_{n_1}^{(1)}$.

Proof. We can use the job-changing argument to show statement (i), which also follows from the fact that the weighted shortest processing time (WSPT) rule is the unique optimal strategy for minimizing the total weighted completion time.

We can also use the job-changing argument to show statement (ii) because, from Lemma 3.1(i), $i < j$ implies that $p_i^{(1)} \leq p_j^{(1)}$ and $w_i^{(1)} \geq w_j^{(1)}$. The lemma follows. □

Lemma 3.4. *For each Pareto-optimal point (C, U, Y), there exists a corresponding Pareto-optimal valid-schedule σ such that the early A_2-jobs are scheduled in their job-index order.*

Proof. We can easily prove the lemma by the job-shifting argument. □

Lemma 3.5. *For each Pareto-optimal point (C, U, Y), there exists a corresponding Pareto-optimal valid-schedule σ such that the following statements hold for the A_3-jobs.*

(i) For each reversed pair of non-late A_3-jobs $J_j^{(3)} J_i^{(3)}$ in σ, we have $w_i^{(3)} < w_j^{(3)}$ and $d_i^{(3)} < d_j^{(3)} < C_i^{(3)}(\sigma)$.

(ii) Each deferred A_3-job is partially early.

(iii) For each non-late A_3-job $J_j^{(3)}$, at most one deferred A_3-job with a job index less than j is scheduled after $J_j^{(3)}$.

(iv) The early A_3-jobs are scheduled in their job-index order. Moreover, the early A_3-jobs with a common due date are consecutively scheduled.

(v) For each deferred A_3-job $J_i^{(3)}$, the A_3-jobs $J_j^{(3)}$ with $j > i$ and $J_j^{(3)} \prec_\sigma J_i^{(3)}$ are strictly early and are consecutively scheduled directly before $J_i^{(3)}$. Consequently, each deferred A_3-job is scheduled immediately after some early A_3-job with a smaller job index.

(vi) If $d_j^{(3)} \uparrow\downarrow w_j^{(3)}$ is a condition in the β-field, then the non-late A_3-jobs are scheduled in their job-index order, and the non-late A_3-jobs with a common due date are consecutively scheduled.

Proof. Let σ be a Pareto-optimal valid-schedule corresponding to (C, U, Y). We use $\mathcal{N}_\sigma^{(3)}$ to denote the set of non-late A_3-jobs in σ. For each job $J_j^{(3)} \in \mathcal{N}_\sigma^{(3)}$, we use $\sigma[J_j^{(3)}]$ to denote the position index of job $J_j^{(3)}$ in σ, i.e., $\sigma[J_j^{(3)}] = i$ if and only if $J_{\sigma(i)} = J_j^{(3)}$. We use $\Phi(\sigma)$ to denote the sum of the position indices of the non-late A_3-jobs in σ and $\Psi(\sigma)$ to denote the number of reversed pairs of non-late A_3-jobs in σ. For our purpose, we may choose such a Pareto-optimal valid-schedule σ corresponding to (C, U, Y) such that $\Psi(\sigma) - \Phi(\sigma)$ is as small as possible, subject to the restriction that $|\mathcal{N}_\sigma^{(3)}|$ is as small as possible. We show in the following that σ satisfies all the statements in this lemma.

Note that if $J_j^{(3)} J_i^{(3)}$ is a reversed pair of non-late A_3-jobs in σ, then

$$i < j,\ d_i^{(3)} \leq d_j^{(3)},\ J_j^{(3)} \prec_\sigma J_i^{(3)}, \text{ and } S_i^{(3)}(\sigma) < d_i^{(3)}. \tag{8}$$

From (4) and (5), the conditions "$i < j$ and $w_i^{(3)} < w_j^{(3)}$" imply "$d_i^{(3)} < d_j^{(3)}$". Then statement (i) is equivalent to the following statement (i′).

(i′) For each reversed pair of non-late A_3-jobs $J_j^{(3)} J_i^{(3)}$ in σ, we have $w_i^{(3)} < w_j^{(3)}$ and $d_j^{(3)} < C_i^{(3)}(\sigma)$.

We prove statement (i′) by the job-shifting argument. If σ violates statement (i′), we pick a reversed pair of non-late A_3-jobs $J_j^{(3)} J_i^{(3)}$ with either $w_i^{(3)} \geq w_j^{(3)}$ or $d_j^{(3)} \geq C_i^{(3)}(\sigma)$ such that $\sigma[J_i^{(3)}] - \sigma[J_j^{(3)}]$ is as small as possible. Then there is no other A_3-job $J_k^{(3)}$ such that $J_j^{(3)} \prec_\sigma J_k^{(3)} \prec_\sigma J_i^{(3)}$. Let σ' be the schedule obtained from σ by shifting job $J_j^{(3)}$ to the position directly after job $J_i^{(3)}$. Moreover, if $J_j^{(3)}$ is late in σ', we define σ'' as the schedule obtained from σ' by removing $J_j^{(3)}$, which leads to σ'' being a valid-schedule such that

$$(\gamma^{(1)}(\sigma''), \gamma^{(2)}(\sigma''), \gamma^{(3)}(\sigma'')) \preceq (\gamma^{(1)}(\sigma'), \gamma^{(2)}(\sigma'), \gamma^{(3)}(\sigma')). \tag{9}$$

Note that

$$C_j^{(3)}(\sigma') = C_i^{(3)}(\sigma) \text{ and } C_i^{(3)}(\sigma') = C_i^{(3)}(\sigma) - p_j^{(3)}. \tag{10}$$

If $w_i^{(3)} \geq w_j^{(3)}$, from (8) and (10), we have $w_i^{(3)} Y_i^{(3)}(\sigma') + w_j^{(3)} Y_j^{(3)}(\sigma') \leq w_i^{(3)} Y_i^{(3)}(\sigma) + w_j^{(3)} Y_j^{(3)}(\sigma)$. If $d_j^{(3)} \geq C_i^{(3)}(\sigma)$, from (10), $J_j^{(3)}$ is early in σ', so we still have $w_i^{(3)} Y_i^{(3)}(\sigma') + w_j^{(3)} Y_j^{(3)}(\sigma') \leq w_i^{(3)} Y_i^{(3)}(\sigma) + w_j^{(3)} Y_j^{(3)}(\sigma)$. For each job $J_z \in \mathcal{J} \setminus \{J_i^{(3)}, J_j^{(3)}\}$, we clearly have $C_z(\sigma') \leq C_z(\sigma)$. This implies that $(\gamma^{(1)}(\sigma'), \gamma^{(2)}(\sigma'), \gamma^{(3)}(\sigma')) \preceq (\gamma^{(1)}(\sigma), \gamma^{(2)}(\sigma), \gamma^{(3)}(\sigma))$, so σ' is a Pareto-optimal schedule corresponding to (C, U, F), too. But then, if $J_j^{(3)}$ is late in σ', from (9), σ'' is a Pareto-optimal valid-schedule corresponding to (C, U, F) such that $|\mathcal{N}_{\sigma''}^{(3)}| < |\mathcal{N}_\sigma^{(3)}|$, contradicting the choice of σ; and if $J_j^{(3)}$ is non-late in σ', σ' is a Pareto-optimal valid-schedule corresponding to (C, U, F) such that $|\mathcal{N}_{\sigma'}^{(3)}| = |\mathcal{N}_\sigma^{(3)}|$, $\Phi(\sigma') \geq \Phi(\sigma)$ and $\Psi(\sigma') < \Psi(\sigma)$, i.e., $\Psi(\sigma') - \Phi(\sigma') < \Psi(\sigma) - \Phi(\sigma)$, contradicting the choice of σ again. Statement (i′) follows and statement (i) follows.

To prove statement (ii), let $J_i^{(3)}$ be a deferred A_3-job in σ. Then there is a non-late A_3-job $J_j^{(3)}$ such that $J_j^{(3)} J_i^{(3)}$ forms a reversed pair in σ. From (8) and statement (i), we have $d_i^{(3)} \leq d_j^{(3)} < C_i^{(3)}(\sigma)$. Then $J_i^{(3)}$ is partially early in σ, as required.

To prove statement (iii), we suppose to the contrary that there are three non-late A_3-jobs $J_j^{(3)}$, $J_k^{(3)}$, and $J_{k'}^{(3)}$ in σ, where $J_k^{(3)}$ and $J_{k'}^{(3)}$ are deferred, such that $k, k' < j$ and $J_j^{(3)} \prec_\sigma J_k^{(3)} \prec_\sigma J_{k'}^{(3)}$. Note that $J_j^{(3)} J_k^{(3)}$ is a reversed pair and $J_{k'}^{(3)}$ is a deferred job. From statement (i), we have $d_j^{(3)} < C_k^{(3)}(\sigma)$ and $J_{k'}^{(3)}$ is partially early in σ. Then we have $d_j^{(3)} < C_k^{(3)}(\sigma) \leq S_{k'}^{(3)}(\sigma) < d_{k'}^{(3)} \leq d_j^{(3)}$. This leads to a contradiction and completes the proof of statement (iii).

We now consider statement (iv). From statement (i), there is no reversed pair of early A-jobs in σ. Thus, the early A_3-jobs are scheduled in their job-index order in σ. Let d be the due date of some early A_3-jobs. Let $\mathcal{F} = \{J_{i_1}^{(3)}, J_{i_2}^{(3)}, \ldots, J_{i_k}^{(3)}\}$ be the set of early A_3-jobs with due date d in σ such that $J_{i_1}^{(3)} \prec_\sigma J_{i_2}^{(3)} \prec_\sigma \cdots \prec_\sigma J_{i_k}^{(3)}$. Then $C_{i_k}^{(3)}(\sigma) \leq d$. From statement (i), we have $i_1 < i_2 < \cdots < i_k$ and no other A_3-jobs are scheduled between the jobs of $\mathcal{F}$ in σ. Let $\mathcal{B}$ be the set of valid jobs J_j with $J_j \notin \mathcal{F}$ and $C_j(\sigma) < C_{i_k}^{(3)}(\sigma)$, and let $\mathcal{A}$ be the set of valid jobs J_j with $C_j(\sigma) > C_{i_k}^{(3)}(\sigma)$. Then $(\mathcal{B}, \mathcal{F}, \mathcal{A})$ forms a partition of the valid jobs in σ. If the jobs in $\mathcal{F}$ are not consecutively scheduled in σ, we define σ' as a new schedule obtained from σ by re-scheduling the valid jobs in σ in the

order $\vec{\mathcal{B}}_\sigma \prec_{\sigma'} \vec{\mathcal{F}}_\sigma \prec_{\sigma'} \vec{\mathcal{A}}_\sigma$. In schedule σ', the jobs of $\mathcal{B}$ are consecutively scheduled from time 0 to time $C^{(3)}_{i_k}(\sigma) - p(\mathcal{F})$ in the same order in σ, the jobs of $\mathcal{F}$ are consecutively scheduled from time $C^{(3)}_{i_k}(\sigma) - p(\mathcal{F})$ to $C^{(3)}_{i_k}(\sigma)$, and the schedule for the jobs of $\mathcal{A}$ is unchanged. Clearly, the A_3-jobs of $\mathcal{F}$ are also early in σ' and $C_z(\sigma') \leq C_z(\sigma)$ for all jobs $J_z \in \mathcal{J} \setminus \mathcal{F}$. This means that σ' is a Pareto-optimal valid-schedule corresponding to (C, U, Y), too. But then, we have $|\mathcal{N}^{(3)}_{\sigma'}| = |\mathcal{N}^{(3)}_{\sigma}|$, $\Phi(\sigma') > \Phi(\sigma)$, and $\Psi(\sigma') = \Psi(\sigma)$, i.e., $\Psi(\sigma') - \Phi(\sigma') < \Psi(\sigma) - \Phi(\sigma)$, contradicting the assumption for σ. Thus, the jobs in $\mathcal{F}$ are consecutively scheduled. Statement (iv) follows.

To prove statement (v), let $J^{(3)}_i$ be a deferred A_3-job in σ. From statement (ii), $J^{(3)}_i$ is partially early in σ. We use $\mathcal{F}$ to denote the set of A_3-jobs $J^{(3)}_j$ with $j > i$ and $J^{(3)}_j \prec_\sigma J^{(3)}_i$. Since $J^{(3)}_i$ is deferred in σ, the set $\mathcal{F}$ is not empty. From Lemma 3.2, all the jobs in $\mathcal{F}$ are strictly early in σ. Suppose to the contrary that the A_3-jobs in $\mathcal{F} \cup \{J^{(3)}_i\}$ are not consecutively scheduled in σ. Let $\mathcal{B}$ be the set of valid jobs J_j with $J_j \notin \mathcal{F}$ and $C_j(\sigma) < C^{(3)}_i(\sigma)$, and let $\mathcal{A}$ be the set of valid jobs J_j with $C_j(\sigma) > C^{(3)}_i(\sigma)$. Then $(\mathcal{B}, \mathcal{F}, \{J^{(3)}_i\}, \mathcal{A})$ forms a partition of the valid jobs in σ. Let σ' be the new schedule obtained from σ by re-scheduling the valid jobs in σ consecutively in the order $\vec{\mathcal{B}}_\sigma \prec_{\sigma'} \vec{\mathcal{F}}_\sigma \prec_{\sigma'} J^{(3)}_i \prec_{\sigma'} \vec{\mathcal{A}}_\sigma$. It is easy to see that $C^{(3)}_i(\sigma') = C^{(3)}_i(\sigma)$, so $J^{(3)}_i$ is also partially early in σ'. For each $J^{(3)}_j \in \mathcal{F}$, $J^{(3)}_j J^{(3)}_i$ is a reversed pair in σ'. From Lemma 3.2 again, all the A_3-jobs in $\mathcal{F}$ are strictly early in σ'. For each job $J_z \notin \mathcal{F}$, we have $C_z(\sigma') \leq C_z(\sigma)$ and at least one A-job has an earlier completion time in σ' than that in σ. Then σ' is a Pareto-optimal valid-schedule corresponding to (C, U, Y) such that $|\mathcal{N}^{(3)}_{\sigma'}| = |\mathcal{N}^{(3)}_{\sigma}|$, $\Phi(\sigma') > \Phi(\sigma)$, and $\Psi(\sigma') = \Psi(\sigma)$, i.e., $\Psi(\sigma') - \Phi(\sigma') < \Psi(\sigma) - \Phi(\sigma)$. This contradicts the assumption for σ and proves statement (v).

To prove statement (vi), we suppose that $d^{(3)}_j \uparrow\downarrow w^{(3)}_j$ is a condition in the β-field. From statement (i), there is no deferred A_3-job in σ. Thus, the non-late A_3-jobs are scheduled in their job-index order. Now let d be the due date of some early A_3-jobs, let $J^{(3)}_i$ be the last non-late A_3-job in σ such that $d^{(3)}_i = d$, and let $\mathcal{F}$ be the set of non-late A_3-jobs $J^{(3)}_j$ with due date d such that $J^{(3)}_j \prec_\sigma J^{(3)}_i$. Then $\mathcal{F} \cup \{J^{(3)}_i\}$ consists of all the non-late A_3-jobs with the common due date d. Since $J^{(3)}_i$ is a non-late A_3-job in σ and $\mathcal{F} \prec_\sigma J^{(3)}_i$, we have $C^{(3)}_j(\sigma) \leq S^{(3)}_i(\sigma) < d$ for all the jobs $J^{(3)}_j \in \mathcal{F}$. Thus, all the jobs in $\mathcal{F}$ are early in σ. From statement (iv), the jobs of $\mathcal{F}$ are consecutively scheduled in σ. In the case where $\mathcal{F}$ and $J^{(3)}_i$ are consecutively scheduled in σ, we have nothing to do. Hence, we assume to the contrary that $\mathcal{F}$ and $J^{(3)}_i$ are not consecutively scheduled in σ. Let σ' be the schedule obtained from σ by shifting $\mathcal{F}$ to the position directly before $J^{(3)}_i$. Similar to the discussion in statement (v), we can verify that σ' is a Pareto-optimal valid-schedule corresponding to (C, U, Y) such that $|\mathcal{N}^{(3)}_{\sigma'}| = |\mathcal{N}^{(3)}_{\sigma}|$ and $\Psi(\sigma') - \Phi(\sigma') < \Psi(\sigma) - \Phi(\sigma)$, contradicting the assumption for σ. Statement (vi) follows and the lemma follows. □

Although the above three lemmas are separately proved for the A_1-jobs, A_2-jobs, and A_3-jobs, it is not hard to see that these results are compatible. Then we have the following lemma.

Lemma 3.6. *For each Pareto-optimal point (C, U, Y), there exists a corresponding Pareto-optimal valid-schedule σ such that the properties given in Lemmas 3.3, 3.4, and 3.5 hold simultaneously.*

The properties established in Lemma 3.6 help us understand the structure of Pareto-optimal valid-schedules for problem $1|\beta|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$. For the convenience of algorithm design, we sometimes only

need the weakened form of these properties.

The following lemma, which is simply an observation, helps us estimate the size of a non-dominated set of vectors.

Lemma 3.7. *Let Γ be a set of non-dominated m-vectors in the form $(u_1, u_2, \ldots, u_m)$. Suppose that the u_i-value has q_i choices in Γ for $i = 1, 2, \ldots, m$, which means that the total number of different values that u_i can attain is q_i. Then we have $|\Gamma| \leq \min_{i=1,2,\ldots,m} \frac{q_1 q_2 \cdots q_m}{q_i}$.*

4 Problem $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$

Let $\sigma = (J_{\sigma(1)}, J_{\sigma(2)}, \ldots, J_{\sigma(n')})$ be a valid-schedule for the problem $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$ on instance $\mathcal{J}$. For each $k \in \{1, 2, \ldots, n'\}$, we call $\sigma_k = (J_{\sigma(1)}, J_{\sigma(2)}, \ldots, J_{\sigma(k)})$ a *subschedule* of σ. When there is no risk of confusion, we also write $\sigma_k = \{J_{\sigma(1)}, J_{\sigma(2)}, \ldots, J_{\sigma(k)}\}$. For each $z \in \{1, 2, 3\}$, we introduce a dummy A_z-job $J_0^{(z)}$ with $p_0^{(z)} = 0$, $d_0^{(z)} = 0$, and $w_0^{(z)} = 0$. Thus, we can sometimes regard $J_0^{(3)}$ as a deferred A_3-job (if necessary) without affecting our discussion. From Lemma 3.6, we have the following lemma.

Lemma 4.1. *For every Pareto-optimal valid-schedule $\sigma = (J_{\sigma(1)}, J_{\sigma(2)}, \ldots, J_{\sigma(n')})$ for the problem $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$, which satisfies the properties in Lemma 3.6, and for every $k \in \{1, 2, \ldots, n'\}$, there is a quadruple (i_1, i_2, i_3, x), where $i_z \in \{1, 2, \ldots, n_z\}$ for $z \in \{1, 2, 3\}$ and $x \in \{0, 1, \ldots, i_3\}$, such that*

(i) the k jobs in $\sigma_k = (J_{\sigma(1)}, J_{\sigma(2)}, \ldots, J_{\sigma(k)})$ consist of all the A_1-jobs of $\mathcal{J}_{i_1}^{(1)}$, the early A_2-jobs of $\mathcal{J}_{i_2}^{(1)}$, and all the non-late A_3-jobs of $\mathcal{J}_{i_3}^{(3)} \setminus \{J_x^{(3)}\}$;

(ii) the A_1-jobs in σ_k are scheduled in their job-index order in σ_k;

(iii) the early A_2-jobs in σ_k are scheduled in their job-index order in σ_k;

(iv) if $x \neq 0$, then $C_{\sigma(k)}(\sigma) < d_x^{(3)}$; all the non-late A_3-jobs $J_j^{(3)}$ with $x + 1 \leq j \leq i_3$ are early in σ_k; and for the last early A_3-job $J_{j^*}^{(3)}$ in σ with $j^* \geq x + 1$ and $J_{j^*}^{(3)} \prec J_x^{(3)}$, $J_{j^*}^{(3)}$ is scheduled directly before $J_x^{(3)}$ in σ (which means that $J_x^{(3)}$ will be expected as a deferred job in σ but not included in σ_k);

(v) all the early A_3-jobs in $\mathcal{J}_{i_3}^{(3)}$ are scheduled in their job-index order in σ_k and, for each non-late A_3-job $J_j^{(3)} \in \mathcal{J}_{i_3}^{(3)}$, at most one deferred A_3-job with a job-index less than j is scheduled after $J_j^{(3)}$ in σ_k.

Proof. Let $i_z = \max\{j : J_j^{(z)} \in \{J_0^{(z)}\} \cup \sigma_k\}$ for $z = 1, 2, 3$. From Lemma 3.6, at most one deferred A_3-job with an index smaller than i_3 is scheduled after $J_{i_3}^{(3)}$ in σ. If such a deferred A_3-job does not exist, we define $x = 0$; if such a deferred A_3-job exists, we define $J_x^{(3)}$ as this deferred A_3-job in σ. If $x \neq 0$, we have $C_{\sigma(k)}(\sigma) \leq S_x^{(3)}(\sigma) < d_x^{(3)}$ since $J_x^{(3)}$ is non-late in σ. Note that, in this case, we have $J_x^{(3)} = J_{\sigma(i)}$ for some $i \in \{k + 1, k + 2, \ldots, n'\}$. Now the correctness of this lemma follows from the properties given in Lemma 3.6. □

Note that the properties given in Lemma 4.1 are much weaker than those given in Lemma 3.6. But they are sufficient and convenient for our algorithm design.

Definition 4.1. *For a quadruple (i_1, i_2, i_3, x) with $i_z \in \{1, 2, \ldots, n_z\}$ for $z \in \{1, 2, 3\}$ and $x \in \{0, 1, \ldots, i_3\}$, a schedule for $\mathcal{J}_{i_1}^{(1)} \cup \mathcal{J}_{i_2}^{(2)} \cup (\mathcal{J}_{i_3}^{(3)} \setminus \{J_x^{(3)}\})$ is called an (i_1, i_2, i_3, x)-schedule if it satisfies the properties given in Lemma 4.1 for σ_k. An $(n_1, n_2, n_3, 0)$-schedule is also called a standard schedule for $\mathcal{J}$. For each (i_1, i_2, i_3, x)-schedule π, we call $(\tau(\pi), C(\pi), U(\pi), Y(\pi))$ a state of (i_1, i_2, i_3, x), where $\tau(\pi)$ is the total processing time of the valid-jobs in π, $C(\pi) = \sum_{j=1}^{i_1} w_j^{(1)} C_j^{(1)}(\pi)$ is the total weighted completion time of the A_1-jobs of $\mathcal{J}_{i_1}^{(1)}$ in π, $U(\pi) = \sum_{j=1}^{i_2} w_j^{(2)} U_j^{(2)}(\pi)$ is the weighted number of tardy A_2-jobs of $\mathcal{J}_{i_2}^{(2)}$ in π, and $Y(\pi) = \sum_{J_j^{(3)} \in \mathcal{J}_{i_3}^{(3)} \setminus \{J_x^{(3)}\}} w_j^{(3)} Y_j^{(3)}$ is the total weighted late work of the A_3-jobs of $\mathcal{J}_{i_3}^{(3)} \setminus \{J_x^{(3)}\}$ in π. We use $\Gamma(i_1, i_2, i_3, x)$ to denote the set of all the states of (i_1, i_2, i_3, x), and $\tilde{\Gamma}(i_1, i_2, i_3, x)$ to denote the set of non-dominated vectors in $\Gamma(i_1, i_2, i_3, x)$. Moreover, we define*

$$\Gamma(n_1, n_2, n_3) = \{(C, U, Y) : (\tau, C, U, Y) \in \tilde{\Gamma}(n_1, n_2, n_3, 0) \text{ for some } \tau \geq P_{n_1}^{(1)}\} \tag{11}$$

and let $\tilde{\Gamma}(n_1, n_2, n_3)$ be the set of non-dominated vectors in $\Gamma(n_1, n_2, n_3)$.

The states of (i_1, i_2, i_3, x) are used to iteratively enumerate all the states of $(n_1, n_2, n_3, 0)$, so that we can get the set of all the target vectors of the standard schedules, i.e., the schedules that satisfy Lemma 4.1. Note that for any Pareto-optimal point, there is a schedule, corresponding to this point, that satisfies Lemma 4.1. If we enumerate all the standard schedules, then the set of the target vectors corresponding to these schedules must contain all the pareto-optimal points. The following two lemmas show that all the non-dominated vectors in the set of target vectors of standard schedules constitute the Pareto frontier.

Note that each Pareto-optimal valid-schedule satisfying the properties in Lemma 3.6 for $\mathcal{J}$ is also a standard schedule for $\mathcal{J}$, but the reverse is not necessarily true.

Lemma 4.2. *For each standard schedule π for $\mathcal{J}$, there is a Pareto-optimal valid-schedule σ for $\mathcal{J}$ such that $(C(\sigma), U(\sigma), Y(\sigma)) \preceq (C(\pi), U(\pi), Y(\pi))$.*

The following lemma reveals the structure of the Pareto frontier of problem $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$.

Lemma 4.3. *The Pareto frontier of problem $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$ on instance $\mathcal{J}$ is $\tilde{\Gamma}(n_1, n_2, n_3)$.*

Proof. From Definition 4.1, $\tilde{\Gamma}(n_1, n_2, n_3)$ is a set of non-dominated vectors, each of which is an objective vector. Then it suffices to show that the Pareto frontier of the considered problem is a subset of $\tilde{\Gamma}(n_1, n_2, n_3)$.

To this end, let (C, U, Y) be a Pareto-optimal point of the considered problem. Let π be a Pareto-optimal valid-schedule for $\mathcal{J}$ with $(C(\pi), U(\pi), Y(\pi)) = (C, U, Y)$ such that $\tau = \tau(\pi)$ is as small as possible. Since π is also a standard schedule for $\mathcal{J}$, we have $(\tau(\pi), C(\pi), U(\pi), Y(\pi)) \in \Gamma(n_1, n_2, n_3, 0)$. Then there is a standard schedule $\tilde{\pi}$ for $\mathcal{J}$ such that $(\tau(\tilde{\pi}), C(\tilde{\pi}), U(\tilde{\pi}), Y(\tilde{\pi})) \preceq (\tau(\pi), C(\pi), U(\pi), Y(\pi))$ and $(C(\tilde{\pi}), U(\tilde{\pi}), Y(\tilde{\pi})) \in \tilde{\Gamma}(n_1, n_2, n_3)$. From Lemma 4.2, there is a Pareto-optimal valid-schedule σ for $\mathcal{J}$ such that $(C(\sigma), U(\sigma), Y(\sigma)) \preceq (C(\tilde{\pi}), U(\tilde{\pi}), Y(\tilde{\pi}))$. Thus, we have

$$(C(\sigma), U(\sigma), Y(\sigma)) \preceq (C(\tilde{\pi}), U(\tilde{\pi}), Y(\tilde{\pi})) \preceq (C(\pi), U(\pi), Y(\pi)). \tag{12}$$

Since both σ and π are Pareto-optimal, the relation in (12) implies that $(C(\sigma), U(\sigma), Y(\sigma)) = (C(\tilde{\pi}), U(\tilde{\pi}), Y(\tilde{\pi})) = (C(\pi), U(\pi), Y(\pi))$. Then we have $(C, U, Y) = (C(\tilde{\pi}), U(\tilde{\pi}), Y(\tilde{\pi})) \in \tilde{\Gamma}(n_1, n_2, n_3)$. The lemma follows. □

From Lemma 4.3, to solve problem $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$, we first generate all the non-dominated state sets $\tilde{\Gamma}(i_1, i_2, i_3, x)$ for all the possible choices of the quadruple (i_1, i_2, i_3, x), and then generate the Pareto-frontier $\tilde{\Gamma}(n_1, n_2, n_3)$. This is implemented by a forward dynamic programming algorithm, with the minor exception that the sets $\Gamma(i_1, i_2, i_3, x)$ with $x > 0$ are generated before the set $\Gamma(i_1, i_2, i_3, 0)$.

Initially, we set $\Gamma(0, 0, 0, 0) := \{(0, 0, 0, 0)\}$ and set $\Gamma(i_1, i_2, i_3, x) := \emptyset$ if $(i_1, i_2, i_3, x) \neq (0, 0, 0, 0)$. Then we recursively generate all the state sets $\Gamma(i_1, i_2, i_3, x)$. To this end, we consider a state $(\tau, C, U, Y) \in \Gamma(i_1, i_2, i_3, x)$, if any, and let π be an (i_1, i_2, i_3, x)-schedule corresponding to the state (τ, C, U, Y). Since $J_x^{(3)}$ is expected to be a deferred job, from Lemma 4.1(iv), we should require $\tau < d_x^{(3)}$. We next generate a state $(\tau', C', U', Y') \in \Gamma(i_1', i_2', i_3', x')$ for some quadruple (i_1', i_2', i_3', x') with $(i_1, i_2, i_3) \prec (i_1', i_2', i_3')$ and $(i_1' - i_1) + (i_2' - i_2) + (i_3' - i_3) = 1$, where $x' \in \{0, x, i_3'\}$. Let π' be an (i_1', i_2', i_3', x')-schedule corresponding to the state (τ', C', U', Y'). We distinguish the following cases.

Case 1. $(i_1', i_2', i_3', x') = (i_1 + 1, i_2, i_3, x)$. Then π' is obtained from π by scheduling $J_{i_1+1}^{(1)}$ directly after schedule π. Thus, we have

$$(\tau', C', U', Y') = (\tau + p_{i_1+1}^{(1)}, C + w_{i_1+1}^{(1)}(\tau + p_{i_1+1}^{(1)}), U, Y).$$

Note that, in this case, if $x \neq 0$, the condition $\tau' < d_x^{(3)}$, i.e., $\tau < d_x^{(3)} - p_{i_1+1}^{(1)}$, must be satisfied. This is deduced from Lemma 4.1(iv).

Case 2. $(i_1', i_2', i_3', x') = (i_1, i_2 + 1, i_3, x)$ and $J_{i_2+1}^{(2)}$ is a tardy A_2-job. Then $\pi' = \pi$ and

$$(\tau', C', U', Y') = (\tau, C, U + w_{i_2+1}^{(2)}, Y).$$

Case 3. $(i_1', i_2', i_3', x') = (i_1, i_2 + 1, i_3, x)$ and $J_{i_2+1}^{(2)}$ is an early A_2-job. Then π' is obtained from π by scheduling $J_{i_2+1}^{(2)}$ directly after schedule π. Thus,

$$(\tau', C', U', Y') = (\tau + p_{i_2+1}^{(2)}, C, U, Y).$$

Note that, in this case, the condition $\tau + p_{i_2+1}^{(2)} \leq d_{i_2+1}^{(2)}$, i.e., $\tau \leq d_{i_2+1}^{(2)} - p_{i_2+1}^{(2)}$, must be satisfied. Moreover, from Lemma 4.1(iv), if $x \neq 0$, the other condition $\tau' < d_x^{(3)}$, i.e., $\tau < d_x^{(3)} - p_{i_2+1}^{(2)}$, must be satisfied, too.

Case 4. $(i_1', i_2', i_3', x') = (i_1, i_2, i_3 + 1, x)$ and $J_{i_3+1}^{(3)}$ is a late A_3-job. Then $\pi' = \pi$ and

$$(\tau', C', U', Y') = (\tau, C, U, Y + w_{i_3+1}^{(3)} p_{i_3+1}^{(3)}).$$

Case 5. $(i_1', i_2', i_3', x') = (i_1, i_2, i_3 + 1, x)$, $x \neq 0$, and $J_{i_3+1}^{(3)}$ is a non-late A_3-job. Then π' is obtained from π by scheduling $J_{i_3+1}^{(3)}$ directly after schedule π. According to Lemma 4.1(iv), $J_{i_3+1}^{(3)}$ is an early job in π'. Thus,

$$(\tau', C', U', Y') = (\tau + p_{i_3+1}^{(3)}, C, U, Y).$$

Since $J^{(3)}_{i_3+1}$ is an early job in π' and $J^{(3)}_x$ is a deferred job, the condition $\tau' < d^{(3)}_x$, i.e., $\tau < d^{(3)}_x - p^{(3)}_{i_3+1}$ must be satisfied.

Case 6. $(i'_1, i'_2, i'_3, x') = (i_1, i_2, i_3+1, x)$, $x = 0$, and $J^{(3)}_{i_3+1}$ is a non-late A_3-job. Then π' is obtained from π by scheduling $J^{(3)}_{i_3+1}$ directly after schedule π. Note that $w^{(3)}_{i_3+1} Y^{(3)}_{i_3+1}(\pi') = w^{(3)}_{i_3+1} \max\{\tau + p^{(3)}_{i_3+1} - d^{(3)}_{i_3+1}, 0\}$. Thus,

$$(\tau', C', U', Y') = (\tau + p^{(3)}_{i_3+1}, C, U, Y + w^{(3)}_{i_3+1} \max\{\tau + p^{(3)}_{i_3+1} - d^{(3)}_{i_3+1}, 0\}).$$

Note that, in this case, the condition $\tau < d^{(3)}_{i_3+1}$ must be satisfied.

Case 7. $(i'_1, i'_2, i'_3, x') = (i_1, i_2, i_3+1, 0)$, $x \neq 0$, and $J^{(3)}_{i_3+1} J^{(3)}_x$ forms a reversed pair in π'. Then π' is obtained from π by scheduling $J^{(3)}_{i_3+1} J^{(3)}_x$ directly after schedule π. In this case, the condition $\tau < d^{(3)}_x - p^{(3)}_{i_3+1}$ must be satisfied to make sure that $J^{(3)}_{i_3+1}$ is early and $J^{(3)}_x$ is non-late. Note that $w^{(3)}_x Y^{(3)}_x(\pi') = w^{(3)}_x \max\{\tau + p^{(3)}_{i_3+1} + p^{(3)}_x - d^{(3)}_x, 0\}$. Thus,

$$(\tau', C', U', Y') = (\tau + p^{(3)}_{i_3+1} + p^{(3)}_x, C, U, Y + w^{(3)}_x \max\{\tau + p^{(3)}_{i_3+1} + p^{(3)}_x - d^{(3)}_x, 0\}).$$

We remark that the case where "$(i'_1, i'_2, i'_3, x') = (i_1, i_2, i_3+1, 0)$, $x \neq 0$, and $J^{(3)}_x \prec_{\pi'} J^{(3)}_{i_3+1}$" is covered in our situation classification.

Case 8. $(i'_1, i'_2, i'_3, x') = (i_1, i_2, i_3+1, i_3+1)$ and $x = 0$. Then $\pi' = \pi$ and $J^{(3)}_{i_3+1}$ is the deferred and unscheduled A_3-job. Thus,

$$(\tau', C', U', Y') = (\tau, C, U, Y).$$

Note that, in this case, the condition $\tau < d^{(3)}_{i_3+1}$ must be satisfied since $J^{(3)}_{i_3+1}$ is deferred.

The above analysis enables us to solve problem $1|p^{(1)}_j \uparrow\downarrow w^{(1)}_j|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$ by the following Algorithm 1.

Algorithm 1: For solving $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$

Set $\Gamma(0,0,0,0) := \{(0,0,0,0)\}$ and set $\Gamma(i_1, i_2, i_3, x) := \emptyset$ if $(i_1, i_2, i_3, x) \neq (0,0,0,0)$.
for $i_1 = 0, 1, \ldots, n_1$, $i_2 = 0, 1, \ldots, n_2$, $i_3 = 0, 1, \ldots, n_3$, *and* $x = 0, 1, \ldots, i_3$, **do**
  for *each* $(\tau, C, U, Y) \in \Gamma(i_1, i_2, i_3, x)$, **do**
    if "$i_1 < n_1$, $x \neq 0$, *and* $\tau < d_x^{(3)} - p_{i_1+1}^{(1)}$" *or* "$i_1 < n_1$ *and* $x = 0$", **then**
      $\Gamma(i_1 + 1, i_2, i_3, x) := \Gamma(i_1 + 1, i_2, i_3, x) \cup (\tau + p_{i_1+1}^{(1)}, C + w_{i_1+1}^{(1)}(\tau + p_{i_1+1}^{(1)}), U, Y)$
    end
    if $i_2 < n_2$, **then**
      $\Gamma(i_1, i_2 + 1, i_3, x) := \Gamma(i_1, i_2 + 1, i_3, x) \cup (\tau, C, U + w_{i_2+1}^{(2)}, Y)$
    end
    if "$i_2 < n_2$, $x \neq 0$, $\tau \leq d_{i_2+1}^{(2)} - p_{i_2+1}^{(2)}$, *and* $\tau < d_x^{(3)} - p_{i_2+1}^{(2)}$" *or* "$i_2 < n_2$, $x = 0$ *and* $\tau \leq d_{i_2+1}^{(2)} - p_{i_2+1}^{(2)}$", **then**
      $\Gamma(i_1, i_2 + 1, i_3, x) := \Gamma(i_1, i_2 + 1, i_3, x) \cup (\tau + p_{i_2+1}^{(2)}, C, U, Y)$
    end
    if $i_3 < n_3$, **then**
      $\Gamma(i_1, i_2, i_3 + 1, x) := \Gamma(i_1, i_2, i_3 + 1, x) \cup (\tau, C, U, Y + w_{i_3+1}^{(3)} p_{i_3+1}^{(3)})$
    end
    if $i_3 < n_3$, $x \neq 0$ *and* $\tau < d_x^{(3)} - p_{i_3+1}^{(3)}$, **then**
      $\Gamma(i_1, i_2, i_3 + 1, x) := \Gamma(i_1, i_2, i_3 + 1, x) \cup (\tau + p_{i_3+1}^{(3)}, C, U, Y)$;
      $\Gamma(i_1, i_2, i_3 + 1, 0) :=$
      $\Gamma(i_1, i_2, i_3 + 1, 0) \cup (\tau + p_{i_3+1}^{(3)} + p_x^{(3)}, C, U, Y + w_x^{(3)} \max\{\tau + p_{i_3+1}^{(3)} + p_x^{(3)} - d_x^{(3)}, 0\})$
    end
    if $i_3 < n_3$, $x = 0$ *and* $\tau < d_{i_3+1}^{(3)}$, **then**
      $\Gamma(i_1, i_2, i_3 + 1, i_3 + 1) := \Gamma(i_1, i_2, i_3 + 1, i_3 + 1) \cup (\tau, C, U, Y)$;
      $\Gamma(i_1, i_2, i_3 + 1, 0) :=$
      $\Gamma(i_1, i_2, i_3 + 1, 0) \cup (\tau + p_{i_3+1}^{(3)}, C, U, Y + w_{i_3+1}^{(3)} \max\{\tau + p_{i_3+1}^{(3)} - d_{i_3+1}^{(3)}, 0\})$
    end
  end
  For each newly generated $\Gamma(i_1', i_2', i_3', x')$, set $\Gamma(i_1', i_2', i_3', x') := \tilde{\Gamma}(i_1', i_2', i_3', x')$
end
Generate $\tilde{\Gamma}(n_1, n_2, n_3)$ and, for each state $(C, U, Y) \in \tilde{\Gamma}(n_1, n_2, n_3)$, derive the corresponding optimal schedule by backtracking.

Note that the upper bounds of $\gamma^{(1)}, \gamma^{(2)}$, and $\gamma^{(3)}$ are given by $Q^{(1)} = \sum_{j=1}^{n_1} w_j^{(1)} p(\mathcal{J})$, $Q^{(2)} = \sum_{j=1}^{n_2} w_j^{(2)}$, and $Q^{(3)} = \sum_{j=1}^{n_3} w_j^{(3)} p_j^{(3)}$, respectively.

Theorem 4.1. *Algorithm 1 solves the Pareto-frontier scheduling problem* $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$ *in* $O(n_1 n_2 n_3^2 Q^{(1)} Q^{(2)} Q^{(3)})$ *time.*

Proof. The correctness of Algorithm 1 is guaranteed by Lemma 4.3. In the following we analyze its time complexity.

The initialization step takes $O(n_1 n_2 n_3^2)$ time, which is dominated by the final time complexity of Algorithm 1.

In the implementation of Algorithm 1, we guarantee that $\Gamma(i_1, i_2, i_3, x) = \tilde{\Gamma}(i_1, i_2, i_3, x)$, which consists of some non-dominated states. There are a total $O(n_1 n_2 n_3^2)$ distinct state sets $\Gamma(i_1, i_2, i_3, x)$ for $i_1 = 0, 1, \ldots, n_1$, $i_2 = 0, 1, \ldots, n_2$, $i_3 = 0, 1, \ldots, n_3$, and $x = 0, 1, \ldots, i_3$. From Lemma 3.7, each state set $\Gamma(i_1, i_2, i_3, x)$ contains $O(Q^{(1)} Q^{(2)} Q^{(3)})$ states in each iteration of Algorithm 1. Moreover, for each state $(\tau, C, U, Y) \in \Gamma(i_1, i_2, i_3, x)$, Algorithm 1 generates at most eight (a constant) new states. Thus, the overall running time of Algorithm 1 is $O(n_1 n_2 n_3^2 Q^{(1)} Q^{(2)} Q^{(3)})$. □

Corollary 4.1. *Problem* $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}, d_j^{(3)} \uparrow\downarrow w_j^{(3)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$ *is solvable in* $O(n_1 n_2 n_3 Q^{(1)} Q^{(2)} Q^{(3)})$ *time.*

Proof. From Lemma 3.5(vi), we know that, for each Pareto-optimal point (C, U, Y) of problem $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}, d_j^{(3)} \uparrow\downarrow w_j^{(3)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$ on instance $\mathcal{J}$, there exists a corresponding Pareto-optimal valid-schedule σ such that the non-late A_3-jobs are scheduled in their job-index order, so no deferred A_3-job exists in σ. Thus, we may fix $x = 0$ in the implementation of Algorithm 1. From the proof of Theorem 4.1, the time complexity can be reduced to $O(n_1 n_2 n_3 Q^{(1)} Q^{(2)} Q^{(3)})$. □

The time complexity of Algorithm 1 is in fact estimated by the number of choices for the variables in $\{i_1, i_2, i_3, x, C, U, Y\}$. The result in Corollary 4.1 is valid since the variable x can be omitted. If we only consider the criteria $\gamma^{(1)}$ and $\gamma^{(2)}$, then the variables i_3, x, and Y can be omitted. From Theorem 4.1, we have the following corollary.

Corollary 4.2. *Problem* $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)}, \gamma^{(2)})$ *is solvable in* $O(n_1 n_2 Q^{(1)} Q^{(2)})$ *time.*

If we only consider the criteria $\gamma^{(1)}$ and $\gamma^{(3)}$, then the variables i_2 and U can be omitted. From Theorem 4.1 and Corollary 4.1, we have the following corollary.

Corollary 4.3. *Problem* $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}|(\gamma^{(1)}, \gamma^{(3)})$ *is solvable in* $O(n_1 n_3^2 Q^{(1)} Q^{(3)})$ *time, while problem* $1|p_j^{(1)} \uparrow\downarrow w_j^{(1)}, d_j^{(3)} \uparrow\downarrow w_j^{(3)}|(\gamma^{(1)}, \gamma^{(3)})$ *is solvable in* $O(n_1 n_3 Q^{(1)} Q^{(3)})$ *time.*

If we only consider the criteria $\gamma^{(2)}$ and $\gamma^{(3)}$, then there is no need to record the variables i_1 and C. Then we have the following corollary.

Corollary 4.4. *Problem* $1||(\gamma^{(2)}, \gamma^{(3)})$ *on instance* $\mathcal{J}$ *is solvable in* $O(n_2 n_3^2 Q^{(2)} Q^{(3)})$ *time, while problem* $1|d_j^{(3)} \uparrow\downarrow w_j^{(3)}|(\gamma^{(2)}, \gamma^{(3)})$ *on instance* $\mathcal{J}$ *is solvable in* $O(n_2 n_3 Q^{(2)} Q^{(3)})$ *time.*

5 Problem $1|p_j^{(2)} = p^{(2)}, p_j^{(3)} = p^{(3)}|(\sum C_j^{(1)}, \sum U_j^{(2)}, \sum Y_j^{(3)})$

In this section we present a polynomial-time algorithm to solve the Pareto-frontier problem

$$1|p_j^{(2)} = p^{(2)}, p_j^{(3)} = p^{(3)}|(\sum C_j^{(1)}, \sum U_j^{(2)}, \sum Y_j^{(3)}). \tag{13}$$

Although the problem in (13) is a special version of problem $1|(p^{(1)}, w^{(1)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$, Algorithm 1 is not polynomial for this special case. Due to the two conditions $p_j^{(2)} = p^{(2)}$ and $p_j^{(3)} = p^{(3)}$ in the β-filed, the problem in (13) has stronger optimality properties as given in the following lemma.

Lemma 5.1. *For each Pareto-optimal point (C, U, Y) of the problem in (13) on instance $\mathcal{J}$, there exists a corresponding Pareto-optimal valid-schedule σ such that the following statements hold for the jobs in $\mathcal{J}$.*

(i) The A_1-jobs are scheduled in their index order $J_1^{(1)}, J_2^{(1)}, \ldots, J_{n_1}^{(1)}$.

(ii) The early A_2-jobs are given by $J_{u+1}^{(2)}, J_{u+2}^{(2)}, \ldots, J_{n_2}^{(2)}$ and are scheduled in this order.

(iii) There is some $v \in \{0, 1, \ldots, n_3\}$ such that the non-late A_3-jobs are given by $J_{v+1}^{(3)}, J_{v+2}^{(3)}, \ldots, J_{n_3}^{(3)}$ and are scheduled in this order.

Proof. We can prove the result using the pairwise job-interchange argument. □

For a pair of indices u and v with $u \in \{0, 1, \ldots, n_2\}$ and $v \in \{0, 1, \ldots, n_3\}$, we define

$$\mathcal{F}^{(u,v)} = \mathcal{J}^{(1)} \cup (\mathcal{J}^{(2)} \setminus \mathcal{J}_u^{(2)}) \cup (\mathcal{J}^{(3)} \setminus \mathcal{J}_v^{(3)}).$$

We use Problem(u, v) to denote the Pareto-frontier problem

$$1|p_j^{(2)} = p^{(2)}, p_j^{(3)} = p^{(3)}|(\sum C_j^{(1)}, \sum Y_j^{(3)})$$

on instance $\mathcal{F}^{(u,v)}$ under the restriction that all the A_2-jobs in $\mathcal{F}^{(u,v)}$ are early and all the A_3-jobs in $\mathcal{F}^{(u,v)}$ are non-late. We use $\Gamma^{(u,v)}(1, u+1, v+1)$ to denote the Pareto frontier of Problem(u, v). Moreover, we define

$$\Gamma'^{(u,v)}(1, u+1, v+1) = \{(C, u, Y + vp^{(3)}) : (C, Y) \in \Gamma^{(u,v)}(1, u+1, v+1)\}.$$

From Lemma 5.1, we have the following lemma.

Lemma 5.2. *The Pareto frontier of the problem in (13) on instance $\mathcal{J}$ consists of all the non-dominated vectors in the set $\cup_{u \in \{0,1,\ldots,n_2\}, v \in \{0,1,\ldots,n_3\}} \Gamma'^{(u,v)}(1, u+1, v+1)$.*

Based on Lemma 5.2, we next present a polynomial-time algorithm to solve Problem(u, v) for a given pair (u, v) with $u \in \{0, 1, \ldots, n_2\}$ and $v \in \{0, 1, \ldots, n_3\}$.

For convenience sake, we re-define the standard schedules. For a triple (i_1, i_2, i_3) with $i_1 \in \{1, 2, \ldots, n_1\}$, $i_2 \in \{u+1, u+2, \ldots, n_2\}$, and $i_3 \in \{v+1, v+2, \ldots, n_3\}$, we define $\mathcal{J}^{(i_1,i_2,i_3)} = \mathcal{J} \setminus (\mathcal{J}_{i_1-1}^{(1)} \cup \mathcal{J}_{i_2-1}^{(2)} \cup \mathcal{J}_{i_3-1}^{(3)})$. Then we have

$$\mathcal{J}^{(i_1,i_2,i_3)} = \{J_{i_1}^{(1)}, J_{i_1+1}^{(1)}, \ldots, J_{n_1}^{(1)}\} \cup \{J_{i_2}^{(2)}, J_{i_2+1}^{(2)}, \ldots, J_{n_2}^{(2)}\} \cup \{J_{i_3}^{(3)}, J_{i_3+1}^{(3)}, \ldots, J_{n_3}^{(3)}\}.$$

Let $\tau(i_1,i_2,i_3)=p(\mathcal{F}^{(u,v)})-p(\mathcal{J}^{(i_1,i_2,i_3)})$. Then

$$\tau(i_1,i_2,i_3)=\sum_{i=1}^{i_1-1}p_i^{(1)}+\sum_{i=u+1}^{i_2-1}p_i^{(2)}+\sum_{i=v+1}^{i_3-1}p_i^{(3)}.$$

Thus, all the values $\tau(i_1,i_2,i_3)$ can be calculated in $O(n_1n_2n_3)$ time in the preprocessing procedure. A schedule π for $\mathcal{J}^{(i_1,i_2,i_3)}$ is called an (i_1,i_2,i_3)-*schedule* if it satisfies the following properties:

(i) The jobs in $\mathcal{J}^{(i_1,i_2,i_3)}$ are consecutively scheduled from time $\tau(i_1,i_2,i_3)$ to time $p(\mathcal{F}^{(u,v)})=\tau(i_1,i_2,i_3)+p(\mathcal{J}^{(i_1,i_2,i_3)})$;

(ii) The A_1-jobs in $\mathcal{J}^{(1)}\setminus\mathcal{J}^{(1)}_{i_1-1}$ are scheduled in their index order in π;

(iii) The A_2-jobs in $\mathcal{J}^{(2)}\setminus\mathcal{J}^{(2)}_{i_2-1}$ are early in π and are scheduled in their index order in π;

(iv) The A_3-jobs in $\mathcal{J}^{(3)}\setminus\mathcal{J}^{(3)}_{i_3-1}$ are non-late in π and are scheduled in their index order in π.

A $(1,u+1,v+1)$-schedule is also called a *standard schedule* for $\mathcal{F}^{(u,v)}$. Obviously, the start time of a $(1,u+1,v+1)$-schedule is 0. For each (i_1,i_2,i_3)-schedule π, we call $(C(\pi),Y(\pi))$ a state of (i_1,i_2,i_3), where $C(\pi)=\sum_{j=i_1}^{n_1}C_j^{(1)}(\pi)$ is the total completion time of the A_1-jobs of $\mathcal{J}\setminus\mathcal{J}^{(1)}_{i_1-1}$ in π and $Y(\pi)=\sum_{j=i_3}^{n_3}Y_j^{(3)}(\pi)$ is the total late work of the A_3-jobs of $\mathcal{J}\setminus\mathcal{J}^{(3)}_{i_3-1}$ in π. We use $\Gamma^{(u,v)}(i_1,i_2,i_3)$ to denote the set of all the states of (i_1,i_2,i_3) and $\tilde{\Gamma}^{(u,v)}(i_1,i_2,i_3)$ to denote the set of non-dominated vectors in $\Gamma^{(u,v)}(i_1,i_2,i_3)$.

Our algorithm is a backwards dynamic programming algorithm. First, we generate all the non-dominated state sets $\tilde{\Gamma}^{(u,v)}(i_1,i_2,i_3)$ for all the possible choices of the triple (i_1,i_2,i_3), and then we generate the Pareto-frontier $\tilde{\Gamma}^{(u,v)}(1,u+1,v+1)$.

Initially, we set $\Gamma^{(u,v)}(n_1+1,n_2+1,n_3+1):=\{(0,0)\}$ and set $\Gamma^{(u,v)}(i_1,i_2,i_3):=\emptyset$ if $(i_1,i_2,i_3)\neq(n_1+1,n_2+1,n_3+1)$. Then we recursively generate all the state sets $\Gamma^{(u,v)}(i_1,i_2,i_3)$. Let us consider a state $(C,Y)\in\Gamma^{(u,v)}(i_1,i_2,i_3)$, if any, and let π be an (i_1,i_2,i_3)-schedule corresponding to the state (C,Y). We next generate a state $(C',Y')\in\Gamma^{(u,v)}(i_1',i_2',i_3')$ with $(i_1',i_2',i_3')\prec(i_1,i_2,i_3)$ and $(i_1-i_1')+(i_2-i_2')+(i_3-i_3')=1$. Let π' be an (i_1',i_2',i_3')-schedule corresponding to the state (C',Y'). We distinguish the following cases.

Case 1. $(i_1',i_2',i_3')=(i_1-1,i_2,i_3)$. Then π' is obtained from π by scheduling $J^{(1)}_{i_1-1}$ directly before schedule π. Thus, we have

$$(C',Y')=(C+\tau(i_1,i_2,i_3),Y).$$

Case 2. $(i_1',i_2',i_3')=(i_1,i_2-1,i_3)$. Then $J^{(2)}_{i_2-1}$ is an early A_2-job in π', which is obtained from π by scheduling $J^{(2)}_{i_2-1}$ directly before schedule π. Thus, we have

$$(C',Y')=(C,Y).$$

Note that this case only happens when $\tau(i_1,i_2,i_3)\leq d^{(2)}_{i_2-1}$.

Case 3. $(i_1',i_2',i_3')=(i_1,i_2,i_3-1)$. Then $J^{(3)}_{i_3-1}$ is a non-late A_3-job in π', which is obtained from π by scheduling $J^{(3)}_{i_3-1}$ directly before schedule π. Thus, we have

$$(C',Y')=(C,Y+\max\{\tau(i_1,i_2,i_3)-d^{(3)}_{i_3-1},0\}).$$

Note that this only happens when $\tau(i_1, i_2, i_3) < d^{(3)}_{i_3-1} + p^{(3)}$.

The above discussion enables us to solve Problem(u, v) by the following Algorithm 2.

Algorithm 2: For solving Problem(u, v)

Set $\Gamma^{(u,v)}(n_1+1, n_2+1, n_3+1) := \{(0,0)\}$ and set $\Gamma^{(u,v)}(i_1, i_2, i_3) := \emptyset$ if $(i_1, i_2, i_3) \neq (n_1+1, n_2+1, n_3+1)$.
for $i_1 = n_1+1, n_1, \ldots, 1$, $i_2 = n_2+1, n_2, \ldots, u+1$, $i_3 = n_3+1, n_3, \ldots, v+1$, **do**
  for *each* $(C, Y) \in \Gamma^{(u,v)}(i_1, i_2, i_3)$, **do**
    if $i_1 > 1$, **then**
      $\Gamma^{(u,v)}(i_1-1, i_2, i_3) := \Gamma^{(u,v)}(i_1-1, i_2, i_3) \cup (C + \tau(i_1, i_2, i_3), Y)$
    end
    if $i_2 > u+1$ *and* $\tau(i_1, i_2, i_3) \leq d^{(2)}_{i_2-1}$, **then**
      $\Gamma^{(u,v)}(i_1, i_2-1, i_3) := \Gamma^{(u,v)}(i_1, i_2-1, i_3) \cup (C, Y)$
    end
    if $i_3 > v+1$ *and* $\tau(i_1, i_2, i_3) < d^{(3)}_{i_3-1} + p^{(3)}$, **then**
      $\Gamma^{(u,v)}(i_1, i_2, i_3-1) := \Gamma^{(u,v)}(i_1, i_2, i_3-1) \cup (C, Y + \max\{\tau(i_1, i_2, i_3) - d^{(3)}_{i_3-1}, 0\})$
    end
  end
  For each newly generated $\Gamma^{(u,v)}(i'_1, i'_2, i'_3)$, set $\Gamma^{(u,v)}(i'_1, i'_2, i'_3) := \tilde{\Gamma}^{(u,v)}(i'_1, i'_2, i'_3)$
end

Theorem 5.1. *For each pair (u, v) with $u \in \{0, 1, \ldots, n_2\}$ and $v \in \{0, 1, \ldots, n_3\}$, Algorithm 2 solves Problem(u, v) in $O(n_1^3 n_2^2 n_3^2)$ time. Consequently, problem $1|p_j^{(2)} = p^{(2)}, p_j^{(3)} = p^{(3)}|(\sum C_j^{(1)}, \sum U_j^{(2)}, \sum Y_j^{(3)})$ is solvable in $O(n_1^3 n_2^3 n_3^3)$ time.*

Proof. The correctness of Algorithm 2 is ensured by the analysis above. Next we discuss the time complexity of the algorithm.

The initialization step of Algorithm 2 takes $O(n_1 n_2 n_3)$ time, which is dominated by the final time complexity of Algorithm 2.

In the implementation of Algorithm 2, we guarantee that $\Gamma^{(u,v)}(i_1, i_2, i_3) = \tilde{\Gamma}^{(u,v)}(i_1, i_2, i_3)$, which consists of some non-dominated states. There are in total $O(n_1 n_2 n_3)$ distinct state sets $\Gamma^{(u,v)}(i_1, i_2, i_3)$ for $i_1 = n_1 + 1, n_1, \ldots, 1$, $i_2 = n_2 + 1, n_2, \ldots, u + 1$, and $i_3 = n_3 + 1, n_3, \ldots, v + 1$. From Lemma 3.7, the size of each state set $\Gamma^{(u,v)}(i_1, i_2, i_3)$ is upper bounded by the number of C-values with $(C, Y) \in \Gamma^{(u,v)}(i_1, i_2, i_3)$, which is estimated in the following claim.

Claim 1. The number of C-values with $(C, Y) \in \Gamma^{(u,v)}(i_1, i_2, i_3)$ is $O(n_1^2 n_2 n_3)$.

To prove Claim 1, we consider an (i_1, i_2, i_3)-schedule π corresponding to a state $(C, Y) \in \Gamma^{(u,v)}(i_1, i_2, i_3)$. Note that there are $n_1 - i_1 + 1$ A_1-jobs, $n_2 - i_2 + 1$ A_2-jobs, and $n_3 - i_3 + 1$ A_3-jobs scheduled in π. We set $t_1 = n_1 - i_1 + 1$, $t_2 = n_2 - i_2 + 1$, and $t_3 = n_3 - i_3 + 1$. There are two extreme cases for the possible arrangement of π:

Case 1. $J_{i_1}^{(1)} \prec_\pi J_{i_1+1}^{(1)} \prec_\pi \cdots \prec_\pi J_{n_1}^{(1)} \prec_\pi (\mathcal{J}^{(2)} \setminus \mathcal{J}_{i_2-1}^{(2)}) \bigcup (\mathcal{J}^{(3)} \setminus \mathcal{J}_{i_3-1}^{(3)})$;

Case 2. $(\mathcal{J}^{(2)} \setminus \mathcal{J}_{i_2-1}^{(2)}) \bigcup (\mathcal{J}^{(3)} \setminus \mathcal{J}_{i_3-1}^{(3)}) \prec_\pi J_{i_1}^{(1)} \prec_\pi J_{i_1+1}^{(1)} \prec_\pi \cdots \prec_\pi J_{n_1}^{(1)}$.

Note that the contributions of the jobs in $(\mathcal{J}^{(2)} \setminus \mathcal{J}_{i_2-1}^{(2)}) \bigcup (\mathcal{J}^{(3)} \setminus \mathcal{J}_{i_3-1}^{(3)})$ to the value of C is 0 for Case 1, but $t_1(t_2 p^{(2)} + t_3 p^{(3)})$ for Case 2. Then the difference in the values of C for the two extreme cases of π is $t_1 t_2 p^{(2)} + t_1 t_3 p^{(3)}$. Note that, in general, the contribution of the jobs in $\mathcal{J}^{(2)} \setminus \mathcal{J}_{i_2-1}^{(2)}$ to the value of C is not more than $t_1 t_2 p^{(2)}$, and the contribution of the jobs in $\mathcal{J}^{(3)} \setminus \mathcal{J}_{i_3-1}^{(3)}$ to the value of C is not more than $t_1 t_3 p^{(3)}$. Then, for each possible arrangement of π, the value of C is given by $ap^{(2)} + bp^{(3)}$ with $0 \le a \le t_1 t_2$ and $0 \le b \le t_1 t_3$. Thus, there are at most $(t_1 t_2 + 1)(t_1 t_3 + 1)$ possibilities for the value of C. Claim 1 follows by noting that $(t_1 t_2 + 1)(t_1 t_3 + 1) = O(n_1^2 n_2 n_3)$.

From Claim 1, the state set $\Gamma^{(u,v)}(i_1, i_2, i_3)$ contains $O(n_1^2 n_2 n_3)$ states in each iteration of Algorithm 2, which follows from the observation that there are $O(n_1 n_2 n_3)$ choices for the value of C. Moreover, for each state $(C, Y) \in \Gamma^{(u,v)}(i_1, i_2, i_3)$, Algorithm 2 generates at most three (a constant) new states. Thus, the overall running time of Algorithm 2 is $O(n_1^3 n_2^2 n_3^2)$.

Generating $\cup_{u \in \{0,1,\ldots,n_2\}, v \in \{0,1,\ldots,n_3\}} \Gamma'^{(u,v)}(1, u + 1, v + 1)$ takes $O(n_1^3 n_2^3 n_3^3)$ time, which dominates the time of generating all the non-dominated vectors in this set.

Consequently, problem $1|p_j^{(2)} = p^{(2)}, p_j^{(3)} = p^{(3)}|(\sum C_j^{(1)}, \sum U_j^{(2)}, \sum Y_j^{(3)})$ on instance $\mathcal{J}$ is solvable in $O(n_1^3 n_2^3 n_3^3)$ time. □

The time complexity of Algorithm 2 is in fact estimated by the number of choices for the variables in $\{u, v, i_1, i_2, i_3, C\}$. If we only consider the criteria $\sum C_j^{(1)}$ and $\sum Y_j^{(3)}$, then the variables u and i_2 can be omitted. In this case, there are $O(n_1 n_3)$ choices for the value of C. Then we have the following corollary.

Corollary 5.1. *Problem* $1|p_j^{(3)} = p^{(3)}|(\sum C_j^{(1)}, \sum Y_j^{(3)})$ *is solvable in* $O(n_1^2 n_3^2)$ *time.*

For problems $1|p_j^{(2)} = p^{(2)}|(\sum C_j^{(1)}, \sum U_j^{(2)})$ and $1|p_j^{(2)} = p^{(2)}, p_j^{(3)} = p^{(3)}|(\sum U_j^{(2)}, \sum Y_j^{(3)})$, we can solve the corresponding Problem$(u, 0)$ and Problem(u, v) backwards in $O(n)$ time in the following way: the last unscheduled A_2-job has priority to be scheduled if it is early at the current time. Consequently, we have the following result.

Theorem 5.2. *(i) Problem* $1|p_j^{(2)} = p^{(2)}|(\sum C_j^{(1)}, \sum U_j^{(2)})$ *is solvable in* $O(nn_2)$ *time. (ii) Problem* $1|p_j^{(2)} = p^{(2)}, p_j^{(3)} = p^{(3)}|(\sum U_j^{(2)}, \sum Y_j^{(3)})$ *is solvable in* $O(nn_2n_3)$ *time.*

6 Conclusions

We study Pareto-optimization of the three-agent single-machine scheduling problem $1|\beta|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$, where $\gamma^{(1)}(\sigma) = \sum_{j=1}^{n_1} w_j^{(1)} C_j^{(1)}(\sigma)$, $\gamma^{(2)}(\sigma) = \sum_{j=1}^{n_2} w_j^{(2)} U_j^{(2)}(\sigma)$, and $\gamma^{(3)}(\sigma) = \sum_{j=1}^{n_3} w_j^{(3)} Y_j^{(3)}(\sigma)$. For some special versions of the problem, we present the NP-hardness results by using some known results in the literature. For problem $1|p^{(1)} \uparrow\downarrow w^{(1)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$, we provide a pseudo-polynomial-time solution algorithm and discuss various implications. In addition, we show that, for various special versions, the time complexity of our algorithms can be further reduced, even down to polynomial time. Table 1 summarizes our research results.

Some topics can be considered in the future research. First, we note that problem $1|d_j^{(2)} = d^{(2)}, d_j^{(3)} = d^{(3)}|(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)})$, which is binary NP-hard, can be solved in pseudo-polynomial time by using an approach different from ours, since it can be shown that, for each Pareto-optimal point, there exists a corresponding Pareto-optimal valid schedule that is composed of five subschedules, among which the first, third and fifth subschedules comprise the A_1-jobs, while the second and fourth subschedules comprise the A_2-jobs and A_3-jobs, respectively. Second, for problem $1||(\sum T_j^{(1)}, \sum Y_j^{(2)})$, we know that it is binary NP-hard, but its exact computational complexity, i.e., whether it is unary NP-hard or pseudo-polynomially solvable, is still open. Moreover, Pareto-optimization of the single-agent single-machine bi-criteria scheduling problem $1||(f, \sum \tilde{w}_j Y_j)$ is another choice for research, where $f \in \{T_{\max}, \sum w_j C_j, \sum w_i U_j, \sum w_j Y_j\}$, where w_j and $\tilde{w}_j$ are different weights for job J_j.

Acknowledgments

The authors would like to thank the Associate Editor and three anonymous referees for their constructive comments and kind suggestions. This research was supported in part by the NSFC under grant numbers 11671368 and 11771406.

Appendix: An explanation for Lemma 4.1

To explain the significance of Lemma 4.1, we consider a new job $J_x^{(z)}$ in some step of our algorithm in which the subschedules for the jobs of the first i_1 A_1-jobs, the first i_2 A_2-jobs, and the first i_3 A_3-jobs

have been enumerated, where $z \in \{1,2,3\}$, $(i_1,i_2,i_3) \leq (n_1,n_2,n_3)$, and $i_1+i_2+i_3 \leq n-1$. Then $J_x^{(z)} \in \{J_{i_1+1}^{(1)}, J_{i_2+1}^{(2)}, J_{i_3+1}^{(3)}\}$. Note that, for each $z \in \{1,2,3\}$, the A_z-jobs have been numbered by some rule, as described in (2), (3), (4), and (5). If $J_x^{(z)}$ is an A_1-job, we schedule it immediately. If $J_x^{(z)}$ is an A_2-job, we either schedule it as an early job immediately or treat it as a tardy job. However, if $J_x^{(z)} = J_x^{(3)}$ is an A_3-job, we either schedule it as a non-late job (deferred or non-deferred) or treat it as a late job. If $J_x^{(3)}$ is non-deferred, it is scheduled immediately. If $J_x^{(3)}$ is deferred, then we do not schedule it immediately until we meet the job $J_{j^*}^{(3)}$, which will be scheduled immediately before $J_x^{(3)}$ such that $J_x^{(3)}$ is non-late, and then we arrange the reversed pair $J_{j^*}^{(3)}J_x^{(3)}$ together. Note that all the non-late A_3-jobs $J_j^{(3)}$ with $x+1 \leq j \leq j^*$ are early and scheduled in their job-index order. We use an example to further illustrate Lemma 4.1.

Instance I_1: In instance I_1, we have two A_1-jobs, two A_2-jobs, and four A_3-jobs defined as follows:

Table 2: Instance I_1

The job set	$\mathcal{J}^{(A_1)}$		$\mathcal{J}^{(A_2)}$		$\mathcal{J}^{(A_3)}$			
The job	$J_1^{(1)}$	$J_2^{(1)}$	$J_1^{(2)}$	$J_2^{(2)}$	$J_1^{(3)}$	$J_2^{(3)}$	$J_3^{(3)}$	$J_4^{(3)}$
The processing time	1	2	1	1	1	3	1	1
The due date			5	7	4	9	10	10
The weight	1	1	1	1	3	1	3	3

Then $\sigma_8 = (J_1^{(1)}, J_2^{(1)}, J_1^{(3)}, J_1^{(2)}, J_3^{(3)}, J_2^{(2)}, J_4^{(3)}, J_2^{(3)})$ is a Pareto-optimal schedule for the above instance (see Figure 1).

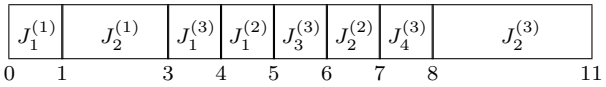

Figure 1: A Pareto-optimal schedule σ_8 corresponding to Instance I_1

According to Lemma 4.1, there are two ways to get σ_8. The first way is to consider the jobs in the following order (note that this is only the order of consideration, not the actual order in the schedule):

$$J_1^{(1)} \prec J_2^{(1)} \prec J_1^{(3)} \prec \underline{J_2^{(3)} \prec J_1^{(2)}} \prec J_3^{(3)} \prec J_2^{(2)} \prec J_4^{(3)}. \tag{14}$$

The second way is to consider the jobs in the following order

$$J_1^{(1)} \prec J_2^{(1)} \prec J_1^{(3)} \prec \underline{J_1^{(2)} \prec J_2^{(3)}} \prec J_3^{(3)} \prec J_2^{(2)} \prec J_4^{(3)}. \tag{15}$$

The quadruple (i_1,i_2,i_3,x) in Lemma 4.1 corresponding to the above two orders (14) and (15) are as follows:

$$(1,0,0,0) \prec (2,0,0,0) \prec (2,0,1,0) \prec \underline{(2,0,2,2) \prec (2,1,2,2)} \prec (2,1,3,2) \prec (2,2,3,2) \prec (2,2,4,0), \tag{16}$$

and

$$(1,0,0,0) \prec (2,0,0,0) \prec (2,0,1,0) \prec \underline{(2,1,1,0) \prec (2,1,2,2)} \prec (2,1,3,2) \prec (2,2,3,2) \prec (2,2,4,0). \tag{17}$$

As the only deferred job in σ_8, $J_2^{(3)}$ is considered as the fourth in (14). At this time, the first three jobs have been arranged, and the numbers of the A_1, A_2, and A_3 jobs we consider are 2, 0, and 2, respectively, while the subscript of the deferred job is 2, so the corresponding quadruple is (2, 0, 2, 2). Note that the deferred job is not scheduled in the current schedule. The reader can imagine putting it somewhere else temporarily. There will be no other deferred job after $J_2^{(3)}$ is deferred until it is scheduled. During the period, we consider and schedule the jobs $J_1^{(2)}, J_3^{(3)}$, and $J_2^{(2)}$, noting that the value of x in all the quadruples is 2. When $J_2^{(3)}$ is scheduled, the value of x in the quadruple becomes 0. Note that both $J_3^{(3)}$ and $J_4^{(3)}$ are early, and $J_4^{(3)}$ is scheduled immediately before $J_2^{(3)}$. The same analysis applies to the order (15).

References

Agnetis, A., Mirchandani, P.B., Pacciarelli, D., & Pacifici, A. (2004). Scheduling problems with two competing agents. *Operations Research*, *52*, 229-242.

Agnetis, A., Billaut, J.C., Gawiejnowicz, S., Pacciarelli, D., & Soukhal, A. (2014). Multiagent scheduling: models and algorithms. Springer.

Agnetis, A., Chen, B., Nicosia, G., & Pacifici, A. (2019). Price of fairness in two-agent single-machine scheduling problems. *European Journal of Operational Research*, *276*, 79-87.

Baker, K.R., & Smith, J.C. (2003). A multiple-criterion model for machine scheduling. *Journal of Scheduling*, *6*, 7-16.

Blazewicz, J. (1984). Scheduling preemptible tasks on parallel processors with information loss. *Technique et Science Informatiques*, *3*, 415-420.

Chen, R.X., & Li, S.S. (2019). Two-agent single-machine scheduling with cumulative deterioration. *4OR-A Quarterly Journal of Operations Research*, *17*, 201-219.

Chen, R.B., Yuan, J.J., & Gao, Y. (2019). The complexity of CO-agent scheduling to minimize the total completion time and total number of tardy jobs. *Journal of Scheduling*, *22*, 581-593.

Cheng, T.C.E., Liu, C.Y., Lee, W.C., & Ji, M. (2014). Two-agent single-machine scheduling to minimize the weighted sum of the agents' objective functions. *Computers & Industrial Engineering*, *78*, 66-73.

Cheng, C.Y., Li, S.F., Ying, K.C., & Liu, Y.H. (2019). Scheduling jobs of two competing agents on a single machine. *IEEE Access*, *7*, 98702-98714.

Choi, B.C., & Chung, J. (2014). Two-agent single-machine scheduling problem with just-in-time jobs. *Theoretical Computer Science*, *543*, 37-45.

Choi, B.C., Chung, J., & Park, M.J. (2019). A just-in-time scheduling problem with two competing agents. *Optimization Letters*. doi: 10.1007/s11590-019-01494-x.

Dover, O., & Shabtay, D. (2016). Single machine scheduling with two competing agents, arbitrary release dates and unit processing times. *Annals of Operations Research*, *238*, 145-178.

Garey, M.R., & Johnson, D.S. (1979). Computers and Intractability: A Guide to the Theory of NP-Completeness. San Francisco: Freeman.

Hariri, A.M.A., Potts, C.N., & Van Wassenhove, L.N. (1995). Single machine scheduling to minimize total weighted late work. *ORSA Journal on Computing*, *7*, 232-242.

Lawler, E.L. (1977). A pseudopolynomial algorithm for sequencing jobs to minimize total tardiness. *Annals of Discrete Mathematics*, *1*, 331-342.

Lawler, E.L., & Moore, J.M. (1969). A functional equation and its applications to resource allocation and sequencing problems. *Management Science*, *16*, 77-84.

Lee, W.C., & Wang, J.Y. (2014). A scheduling problem with three competing agents. *Computers & Operations Research*, *51*, 208-217.

Lee, W.C., & Wang, J.Y. (2017). A three-agent scheduling problem for minimizing the makespan on a single machine. *Computers & Industrial Engineering*, *106*, 147-160.

Li, S.S., & Yuan, J.J. (2020). Single-machine scheduling with multi-agents to minimize total weighted late work. *Journal of Scheduling*, *23*, 497-512.

Moore, J.M. (1968). An n job, one machine sequencing algorithm for minimizing the number of late jobs. *Management Science*, *15*, 102-109.

Oron, D., Shabtay, D., & Steiner, G. (2015). Single machine scheduling with two competing agents and equal job processing times. *European Journal of Operational Research*, *244*, 86-99.

Potts, C.N., & Van Wassenhove, L.N. (1992). Single machine scheduling to minimize total late work. *Operations Research*, *40*, 586-595.

Sahni, S. (1976). Algorithms for scheduling independent tasks. *Journal of the Association of Computing Machinery*, *23*, 116-127.

Smith, W.E. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, *3*, 59-66.

Wan, L., Yuan, J.J., & Wei, L.J. (2016). Pareto optimization scheduling with two competing agents to minimize the number of tardy jobs and the maximum cost. *Applied Mathematics and Computation*, *273*, 912-923.

Wan, L., Mei, J.J., & Du, J.Z. (2020). Two-agent scheduling with unit processing time to minimize total weighted completion time and total weighted number of tardy jobs. *European Journal of Operational Research*, doi: 10.1016/j.ejor.2020.07.064.

Yin, Y., Wang, D.J., Wu, C.C., & Cheng, T.C.E. (2016). CON/SLK due date assignment and scheduling on a single machine with two agents. *Naval Research Logistics*, *63*, 416-429.

Yuan, J.J. (2016). Complexities of some problems on multi-agent scheduling on a single machine. *Journal of the Operations Research Society of China*, *4*, 379-384.

Yuan, J.J. (2017). Multi-agent scheduling on a single machine with a fixed number of competing agents to minimize the weighted sum of number of tardy jobs and makespans. *Journal of Combinatorial Optimization*, *34*, 433-440.

Yuan, J.J. (2018). Complexities of four problems on two-agent scheduling. *Optimization Letters*, *12*, 763-780.

Yuan, J.J., Ng, C.T., & Cheng, T.C.E. (2020). Scheduling with release dates and preemption to minimize multiple max-form objective functions. *European Journal of Operational Research*, *280*, 860-875.

Zhang, X.G., & Wang, Y. (2017). Two-agent scheduling problems on a single-machine to minimize the total weighted late work. *Journal of Combinatorial Optimization*, *33*, 945-955.

Zhang, Y., & Yuan, J.J. (2019). A note on a two-agent scheduling problem related to the total weighted late work. *Journal of Combinatorial Optimization*, *37*, 989-999.