J Optim Theory Appl (2017) 174:927–944
DOI 10.1007/s10957-017-1147-7

Polynomial Time Approximation Scheme for Two Parallel Machines Scheduling with a Common Due Date to Maximize Early Work

Malgorzata Sterna[1] · **Kateryna Czerniachowska**[1]

Received: 2 January 2017 / Accepted: 24 July 2017 / Published online: 2 August 2017

Abstract We study the scheduling problem with a common due date on two parallel identical machines and the total early work criterion. The problem is known to be NP-hard. We prove a few dominance properties of optimal solutions of this problem. Their proposal was inspired by the results of some auxiliary computational experiments. Test were performed with the dynamic programming algorithm and list algorithms. Then, we propose the polynomial time approximation scheme, based on structuring problem input. Moreover, we discuss the relationship between the early work criterion and the related late work criterion. We compare the computational complexity and approximability of scheduling problems with both mentioned objective functions.

Keywords Scheduling · Parallel machines · Early work · Late work · Polynomial time approximation scheme

Mathematics Subject Classification 90B35 · 68Q25

1 Introduction

The scheduling theory provides models and methods helpful in solving practical problems (cf., e.g., [1,2]). To cover different application fields, several performance measures have been proposed in the literature, which allow for modeling various

✉ Malgorzata Sterna
malgorzata.sterna@cs.put.poznan.pl

Kateryna Czerniachowska
kateryna.czerniachowska@cs.put.poznan.pl

[1] Institute of Computing Science, Poznan University of Technology, Piotrowo 2, 60-965 Poznan, Poland

optimization criteria. Criteria involving due dates seem to be especially interesting, because time restrictions are very often met in practice.

Within this paper, we analyze the early work criterion, which maximizes the amount of work executed before the due date (i.e. the total size of early parts of jobs) and the late work criterion, strictly related to this objective function, minimizing late parts of jobs [3]. These performance measures find many practical applications. They can be met, e.g., in control systems when collecting data from sensors [3], in agriculture in the process of harvesting crops or spreading fertilizers/pesticides on fields (cf., e.g., [4,5]), in manufacturing systems in planning technological processes or scheduling customer orders (cf., e.g., [6,7]), or in software engineering in the process of software testing or debugging (cf., [8]).

The considered criteria have been mainly studied for the single machine and for dedicates machines problems. There are relatively few results for the parallel machines. For all mentioned environments, a lot of attention has been paid to the common due date cases. Such basic scheduling models are often met in practice too. They can appear in optimizing the execution of production tasks or any other activities within a given planning horizon (e.g., in constructing a factory production plan for a month). Similarly, we can optimize work assignment to employees within a shift (e.g., manufacturing or assembling parts, packing items ordered by customers in Internet shops). Representing the shift length with a common due date, we can use early/late work models for optimizing labor cost. The employer is interested in minimizing the working time of employee exceeding the contracted period (i.e. the shift length), because it causes additional costs. From this point of view, the sequence of jobs is less important than their assignment before and after the due date, since the payment rate depends on whether work is done within normal working time or in overtime.

Within the presented research, we studied the two parallel identical machines scheduling problem with a common due date to maximize the total early work, which is known to be NP-hard [9]. We proved a few dominance properties of optimal solutions of this problem, which allowed us to propose the polynomial time approximation scheme. These properties were disclosed based on the results of some auxiliary computational experiments performed for dynamic programming [9] and list algorithms. Moreover, we studied the relationship between the late and early work, which has not been addressed formally before. We showed that these two criteria are equivalent when the optimal solutions are considered, but they have different nature, when the existence of approximation algorithms with a bounded approximation ratio is taken into account.

The remainder of this paper is organized as follows. In Sect. 2 we shortly discuss the related results presented in the literature so far. In Sect. 3 we give the formal definition of the problem considered in this paper, and study the relationship between the late work and early work criteria, focusing on approximability issues. Section 4 presents dominance properties proved for the early work problem, inspired by computational experiments results, which are briefly summarized. The polynomial time approximation scheme is proposed in Sect. 5. The paper is concluded in Sect. 6.

2 Related Work

The early work (X) is a complementary criterion to the late work (Y), which has been formulated as the first one, and mainly studied in the literature. It was proposed by Blazewicz [3] for the parallel identical machines environment, but the majority of results available in the literature concerns the single machine problem (cf., e.g., [4,10,11]) and shop systems (cf., e.g., [5,12–17]). The results obtained till ca. 2011 have been collected first by Leung [18] and then by Sterna [8] in the survey papers. They concerned mostly NP-hardness proofs, proposals of pseudopolynomial time methods, and polynomial time algorithms for the simplified models. More recently, some new results have been published in the literature. They concern mainly the application of metaheuristics to late work scheduling problems, but also some new—more complex—scheduling models have been studied. Wu et al. [19] considered a single machine scheduling problem with a position-based learning effect and the total late work criterion, proposing a few simulated annealing approaches. Then, Wu et al. [20] proposed for the same problem a branch and bound algorithm with some dominance rules and three heuristic-based genetic algorithms. The learning effect was taken into account also by Chen et al. [21], who proposed the particle swarm optimization algorithm for the flow shop problem. Zhang and Wang [22] studied two-agent scheduling problems, where different agents share a common resource, namely a single machine, and each agent wants to minimize a cost function depending on its jobs only. They considered the total weighted late work and the maximum cost. Wang et al. [23] considered the two-agent scheduling problem where the goal is to minimize the total late work of the first agent, not exceeding a given value of the maximum lateness of the second agent. They proposed the dynamic programming algorithm, branch and bound method and tabu search approach, depending on the instance size. Piroozfard and Wong [24] performed computational experiments for the non-preemptive job shop problem with late work, comparing simulated annealing and genetic algorithm. Afzalirad and Rezaeian [25] studied the unrelated parallel machine scheduling problem with sequence-dependent set-up times, release dates, machine eligibility, precedence constraints and the total late work, proposing a mixed integer linear programming model and two hybrid metaheuristics, genetic algorithm and ant colony optimization, combined with the acceptance strategy of the simulated annealing algorithm. Abasian et al. [26] considered the problem with two precedence-related jobs, two identical parallel machines, job-dependent communication delay and the weighted late work criterion, proposing an integer linear mathematical programming model and a branch and bound algorithm. Al Zuwaini and Zeyad [27] proposed a branch and bound algorithm for a single machine problem with bi-criteria: total late work and the number of tardy jobs. Ren et al. [7] considered the late work in the manufacturing system, where several suppliers provide component parts to a manufacturer assembling products from all parts delivered. They showed the unary NP-hardness of the suppliers' total late work minimization problem. Ranjbar et al. [28] studied the resource-constrained project scheduling problem with the weighted late work criterion. They considered the model with finish-to-start type precedence relations with zero time lag, and one or more constrained renewable resources. They proposed a linear integer programming model, and presented computational results obtained with CPLEX and a branch and bound algorithm.

As we have mentioned in Sect. 1, there are relatively few results concerning the parallel identical machine environment, studied in this paper, which corresponds to those applications, where machines, factory units, workers, teams, etc. have identical capabilities and speeds. Polynomial time algorithms based on network flow were presented for preemptive problems with an arbitrary number of machines, job release times with the late work criterion ($P|r_j, pmtn|Y$) [18], and with the weighted late work criterion ($P|r_j, pmtn|Y_w$) [3,29]. Similar non-preemptive problems with unit processing times ($Pm|r_j, p_j = 1|Y_w$ and $P|r_j, p_j = 1|Y_w$) are also polynomially solvable [8]. For non-preemptive cases, Blazewicz [3] proved the unary NP-hardness of problem $P||Y$, while Sterna [8] mentioned intractability of problem $P2|p_j = 1, \text{chains}|Y$.

The late work minimization problem is closely related to the early work maximization problem, but these two problems are not fully equivalent, as we will show in the next section. The relationship between those two criteria has not been deeply studied yet. Obviously a schedule, which is optimal for the late work minimization, is optimal for the early work maximization. This observation has been already used in the literature while constructing optimal algorithms for the late work criterion (cf., e.g., [13,15]). However, the polynomial approximation schemes have been proposed only for the late work criterion for the single machine problems (cf., [11,30,31]). They have not been studied for multiple machine problems with early/late work criteria yet.

The presented research strictly relates to the results obtained by Chen et al. [9]. The work by Chen et al. [9], inspired also Xu et al. [32], who reported computational experiments for heuristic approaches proposed for the weighted late work problem, i.e. $P2|d_j = d|Y_w$; they tested a few list scheduling algorithms and metaheuristics (ant colony system, genetic algorithm, and simulated annealing). Chen et al. [9] studied the two-machine problem with a common due date in offline and online modes, taking into account the total late work as well as the total early work criteria. They proved the binary NP-hardness of the offline version of $P2|d_j = d|Y$, where the set of jobs is known in advance, by showing the transformation from the partition problem and proposing a pseudopolynomial time dynamic programming algorithm. They mentioned the unary NP-hardness of problem $P|d_j = d|Y$. In the online version, the set of jobs is unknown in advance, and a new job may appear in the system after scheduling the previous one. For the online model with the total early work criterion, Chen et al. [9] proposed an online algorithm for an arbitrary number of machines, called Extended First Fit (EFF), proving its approximation ratio dependent on the number of machines. Moreover, Chen et al. [9] showed that EFF is an optimal online algorithm for two machines, since its approximation ratio equals the lower bound of this ratio. Chen et al. [9] did not study the approximability of the original offline formulation of problem $P2|d_j = d|Y$.

It is worth to be mentioned that different concept of early work can be also found in the literature. Ben Yehoshua and Mosheiov [33] studied a single machine scheduling problem with the minimum total early work criterion. They minimize, instead of maximizing, the duration of the parts of jobs completed prior to their due dates, analyzing different scheduling models with another application fields from the ones considered within this paper. Moreover, the late/early work scheduling is strictly related to the imprecise computation model (cf., [18] for a survey) and to the

scheduling models with variable processing times (cf., [34] or Chapter 6 in [35] for surveys).

3 Problem Formulation

In this paper we study problem $P2|d_j = d|X$, which requires scheduling a set of jobs $J = \{J_1, \ldots, J_j, \ldots, J_n\}$ on two identical parallel machines M_1 and M_2 in non-overlapping and non-preemptive way. Each job J_j is described by the processing time p_j. The quality of a solution, requiring assigning particular jobs to machines, is estimated with regard to job completion times c_j and the common due date d, which represents the preferred completion time for all jobs ($d_j = d$). We maximize the size of early parts of all jobs, $X_j = \min\{p_j, \max\{0, d-(c_j-p_j)\}\}$, that means the total early work, $X = \sum_{j=1}^{n} X_j$. As it was stressed before, the early work parameter, illustrated in Fig. 1, is a complementary parameter to the late work, which was originally proposed in the literature. The late work is defined as $Y_j = \min\{p_j, \max\{0, c_j - d\}\}$, and the total late work as $Y = \sum_{j=1}^{n} Y_j$. The late work criterion is closely related to the makespan and the tardiness criterion [36], but unlikely those two performance measures this parameter is upper bounded by the job processing time.

Within the remainder of the paper, we will denote with $p_{\text{sum}} = \sum_{j=1}^{n} p_j$ the total processing time of all jobs, with $p_{\max} = \max_{j=1\ldots n}\{p_j\}$ the processing time of the longest job, and with C_k the completion time (i.e. the workload) on machine M_k for $k = 1, 2$. Obviously, the schedule makespan equals $C_{\max} = \max\{C_1, C_2\}$. In the two-machine case considered in this research, the total early work equals $X = \min\{d, C_1\} + \min\{d, C_2\}$.

Obviously determining the optimal total early work X^* is equivalent to determining the optimal total late work Y^*, since $Y^* = p_{\text{sum}} - X^*$, and in any feasible schedule $Y_j = p_j - X_j$ for all jobs. Thus problems $P2|d_j = d|Y$ and $P2|d_j = d|X$ are equivalent, if optimal solutions are considered, but they have different natures, if the quality of approximation is discussed.

Since the problem of late work minimization on two machines, $P2|d_j = d|Y$, is NP-hard [9], the corresponding problem of early work maximization, $P2|d_j = d|X$, is also NP-hard. Moreover, the pseudopolynomial time dynamic programming method proposed for $P2|d_j = d|Y$ by Chen et al. [9], can be used for solving $P2|d_j = d|X$ optimally, and we can classify the latter problem as binary NP-hard. The problems for

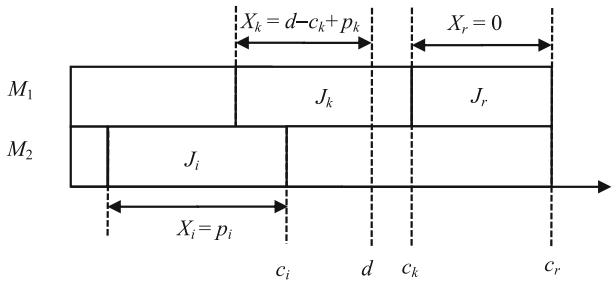

Fig. 1 The early work parameter for early (J_i), partially early (J_k) and late (J_r) jobs

an arbitrary number of machines, $P|d_j = d|X$ and $P|d_j = d|Y$, are already unary NP-hard [9].

On the other hand, the problem with the total late work, $P2|d_j = d|Y$, is non-approximable, due to the nature of this objective function, which takes zero value if all jobs are early. Based on the results by Alon et al. [37], we conclude that there exists no polynomial time approximation algorithm with finite performance guarantee for this problem, particularly no polynomial time approximation scheme (PTAS) exists, unless P = NP. Taking into account that the NP-complete partition problem [38] transforms to the decision version of $P2|d_j = d|Y$ [9], a hypothetical PTAS for this scheduling problem would have solved the partition problem in polynomial time [37]. On the contrary, the problem with the total early work, $P2|d_j = d|X$, is approximable, this means that there exist some approximation algorithms with a finite approximation ratio. Chen et al. [9] have already proposed the list algorithm with the guaranteed performance: Extended First Fit heuristic with the approximation ratio $r_m = (\sqrt{2m^2 - 2m + 1} - 1)/(m - 1)$, where m denotes the number of machines (the method is able to solve more general case with m machines too), and the pseudopolynomial complexity $O(nd^2)$. Within this paper, in Sect. 5, we propose the polynomial time approximation scheme, which constructs schedules with guaranteed $(1 - 3\varepsilon)$ quality in $O(\max\{n, \frac{1}{\varepsilon}2^{\frac{1}{\varepsilon}}\})$ time for any $0 < \varepsilon < 1$. The proposal of PTAS for $P2|d_j = d|X$ allows us to classify this problem as approximable and underlines its different nature in comparison with the non-approximable problem $P2|d_j = d|Y$. Both problems are binary NP-hard, but problem $P2|d_j = d|X$ belongs to the class of problems possessing approximation algorithms, class APX (cf., e.g., [37,38,40]), while $P2|d_j = d|Y$ does not.

4 Dominance Properties

The theoretical studies on properties of problem $P2|d_j = d|X$, crucial for proposing the polynomial time approximation scheme, were preceded by auxiliary computational experiments, which gave some insight in the structure of feasible and optimal solutions of this problem. We implemented the dynamic programming algorithm (DP) proposed by Chen et al. [9], which provided optimal solutions, and four list algorithms. Three of these methods assign jobs to the machine with the minimum workload: in the input (MW), in the longest processing time (LPT), and in the shortest processing time (SPT) order, respectively. The fourth one, Extended First Fit algorithm (EFF) proposed by Chen et al. [9], assigns jobs to the first suitable machine, or to the machine with the minimum workload. The machine is suitable if its workload after assigning a current job will not exceed a bound equal to $r_2 d$, where $r_2 = \sqrt{5} - 1$ denotes the assumed approximation ratio r_m for 2 machines. All methods were implemented in Visual C#, and tested for randomly generated instances on PC with AMD Athlon II X2 245 2.90GHz and 3GB RAM. To analyze the influence of the common due date on the solution process, we performed computational experiments for small instances changing the value of d. For $n \in \{5, 6, \ldots, 20\}$, we generated 10 instances for each instance size with uniform distribution of job processing times $p_j \in [1, 20]$. For each instance, several tests were performed with various due date values $d = qp_{\text{sum}}$,

where $0 < q \leq 1$, namely $q \in \{0.1, 0.15, 0.2, \ldots, 1\}$. Then, we performed similar experiments for larger instances with $n \in \{10, 20, \ldots, 150\}$ for the fixed due date $d = 0.5p_{\text{sum}}$ (the value chosen after the first stage of experiments).

The analysis of optimal solutions obtained for various due dates showed that most instances are easy. Figure 2 presents the ratio (in percent) between the value of the optimal early work, determined by dynamic programming, and the upper bound of the criterion value, determined as $\min\{p_{\text{sum}}, 2d\}$.

For small due dates, there are late jobs on both machines, and the criterion exceeds its maximum value equal to $2d$. Similarly for large due dates, idle time appears before d on both machines, and again the criterion reaches its maximum value equal to p_{sum} in this case. From the optimization point of view, only instances with due dates between ca. 40% and ca. 55% of the total processing time are interesting, because playing with job assignment might improve the criterion value. Thus, the experiments with large instances were performed for $d = 0.5p_{\text{sum}}$ for which the difference between X^* and its upper bound was the biggest. The influence of the due date value and the total (as well as the maximum) processing time on the instance difficulty inspired theoretical studies on dominance properties presented in the remainder of this section.

The computational experiments showed very high efficiency of list algorithms for small instances, except from EFF algorithm. Figure 3 presents the error made by particular algorithms (called average approximation ratio), i.e. the ratio between the optimal early work and the criterion value obtained by these heuristics. EFF's behavior differs from the behavior of the remaining methods, because it is the only approach, which does not assign jobs to the machine with smaller current workload. EFF keeps assigning jobs to the first machine till its workload reaches $r_2 d$, this means it always exceeds the due date, instead of keeping balance between both machines as other methods do. Therefore, the strategy for which the theoretical approximation ratio can be easily determined [9] might be not efficient in practice.

Within the experiments, we wanted to check not only the influence of the due date, but also of the number of jobs, on the instance difficulty and the efficiency of particular list algorithms (cf., Fig. 4). The test results showed that the larger instances are, in general, easier to be solved for all heuristics, except from EFF. When the number of

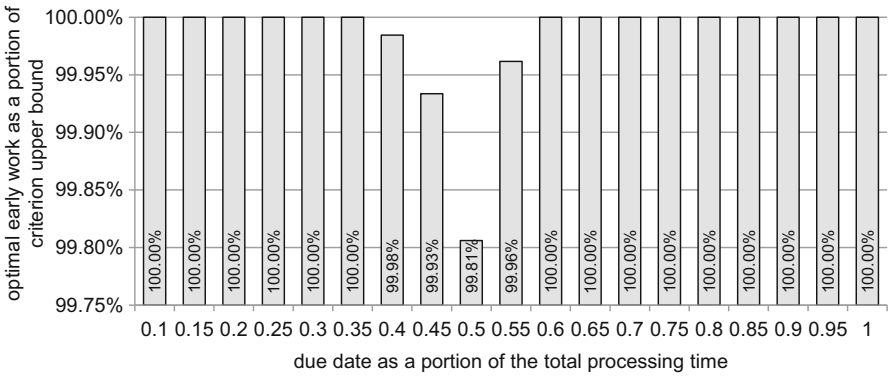

Fig. 2 The ratio between the optimal early work and the upper bound of the early work

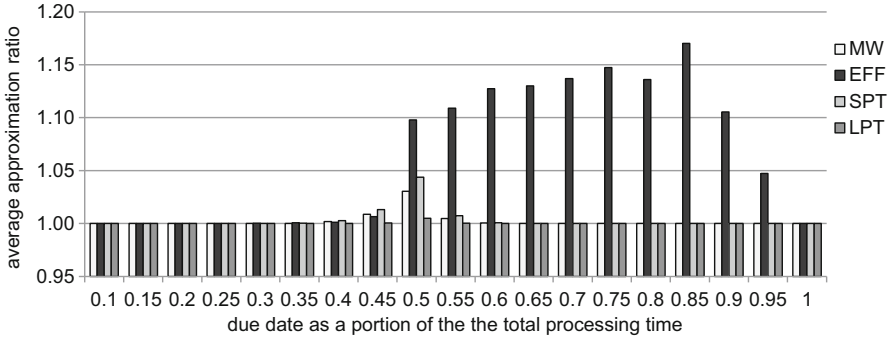

Fig. 3 The average approximation ratio for list algorithms for small instances for particular due dates

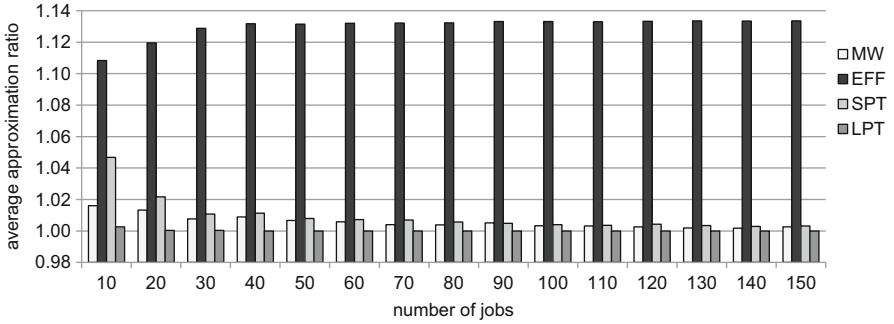

Fig. 4 The average approximation ratio for list algorithms for large instances of different sizes

jobs is large, then the way in which these jobs are assigned to the machines (i.e. the strategy used within a list algorithm) is less crucial than in the case of the small number of jobs. The differences in the criterion value between different schedules for the same instance are relatively less visible than for the instance with the small number of jobs.

The same effect—decrease of the average approximation ratio with the number of jobs for all algorithms except from EFF—is also visible while comparing aggregated results for all small and large instances (cf., Fig. 5). This means that for large instances, simple list heuristics (especially LPT) are very efficient and sufficient. There is no need to construct more sophisticated algorithms, such as metaheuristics, because list algorithms are able to solve the problem optimally in most cases.

The computational experiments showed that many instances of the early work maximization problem are trivial, i.e. any schedule is optimal or an optimal schedule can be determined in polynomial time. If the common due date is very small or very big, then an optimal schedule for these special instances can be constructed without exploring the solution space, despite the fact that the problem is NP-hard. The analysis of those instances resulted in the following dominance properties useful for constructing an approximation scheme.

Property 4.1 *If* $p_{\max} \geq \frac{1}{2} p_{\text{sum}}$, *then* $X^* = \min\{d, p_{\max}\} + \min\{p_{\text{sum}} - p_{\max}, d\}$.

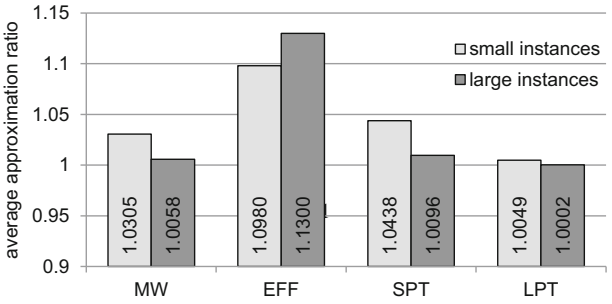

Fig. 5 The average approximation ratio for particular list algorithms for small and large instances

Proof Let's assume that the longest job is scheduled on M_1, and the remaining jobs are scheduled on M_2, then $C_1 = p_{\max} \geq \frac{1}{2} p_{\text{sum}}$ and $C_2 = p_{\text{sum}} - p_{\max} \leq \frac{1}{2} p_{\text{sum}} \leq C_1$. Because jobs are non-preemptive, no unit of work can be shifted from M_1 to M_2, possibly before the common due date d, so the total early work cannot increase, and the schedule is optimal. □

From Property 4.1, we can assume that for non-trivial instances of the problem:

$$p_{\max} < \frac{1}{2} p_{\text{sum}}. \tag{1}$$

Property 4.2 *If $p_{\max} \geq d$, then $X^* = d + \min\{p_{\text{sum}} - p_{\max}, d\}$.*

Proof Let's assume that the longest job is scheduled on M_1, and the remaining jobs are scheduled on M_2, then $C_1 = p_{\max}$ and $C_2 = p_{\text{sum}} - p_{\max}$. The workload on M_1 exceeds the due date and the early work on this machine cannot increase, since it exceeds its maximal value equal to d. The remaining jobs are assigned to M_2, so the early work cannot be increased on this machine either, and the schedule is optimal. □

From Property 4.2, we can assume that for non-trivial instances of the problem:

$$p_{\max} < d \tag{2}$$

and, from (1) and (2), that:

$$\bigvee_{j=1\ldots n} p_j < \frac{1}{2} p_{\text{sum}} \wedge p_j < d. \tag{3}$$

Property 4.3 *If $p_{\text{sum}} \leq d$, then $X^* = p_{\text{sum}}$.*

Proof Any schedule without idle time is optimal, because the makespan cannot exceed the common due date, and all jobs are early. □

From Property 4.3, we can assume that for non-trivial instances of the problem:

$$p_{\text{sum}} > d. \tag{4}$$

Property 4.4 *For any instance of* $P2|d_j = d|X$, $\min\{d, p_{\text{sum}}\} \leq X^* \leq \min\{2d, p_{\text{sum}}\}$.

Proof The upper bound of the total early work is determined by the total processing time p_{sum} (if all jobs are early), or by the doubled due date value $2d$ (if the intervals till the due date on both machines are completely filled with jobs). Taking into account Property 4.3, if $p_{\text{sum}} \leq d$, then the lower bound of the total early work is determined by the total processing time (all jobs are early). If $p_{\text{sum}} > d$, then by executing all jobs on one machine, we can construct a schedule with the total early work equal to d, but a better schedule might be obtained by dividing jobs between machines. □

Taking into account Property 4.4 and (4), we can assume that for non-trivial instances of the problem:

$$d \leq X^* \leq 2d. \tag{5}$$

Property 4.5 *If* $p_{\text{sum}} \geq 3d$, *then* $X^* = 2d$.

Proof Consider the schedule obtained with the longest processing time rule LPT, i.e. by assigning the longest job to the machine with the smaller workload. We claim that this schedule is optimal. If $\min\{C_1, C_2\} \geq d$, then $X^* = 2d$ because the workload exceeds the due date on both machines. Let's assume that $\min\{C_1, C_2\} < d$. We can assume without loss of generality that $C_1 < C_2$, which implies $C_1 < d$ and $C_2 = p_{\text{sum}} - C_1 \geq 3d - C_1 > 3d - d = 2d$, i.e. $C_2 > 2d$. Note, that in the LPT schedule, the difference between workloads on both machines is bounded by the maximum processing time at any stage of this algorithm, i.e. $|C_2 - C_1| \leq p_{\max}$. If before assigning job J_j the workloads on both machines are the same, $C_1 = C_2$, then after assigning this job $|C_2' - C_1'| = p_j \leq p_{\max}$. If before assigning job J_j the workloads on both machines are different, we can assume without loss of generality that $C_1 < C_2$ (i.e. $C_2 - C_1 > 0$), then after assigning this job to machine M_1, we have the difference between workloads $|C_2' - C_1'| = |C_2 - (C_1 + p_j)| = |p_j - (C_2 - C_1)| < p_j \leq p_{\max}$. Based on Property 4.2 and (2), we can assume that $p_{\max} < d$. Hence, in the LPT schedule $|C_2 - C_1| \leq p_{\max} < d$. Since $C_1 < C_2$, we have $|C_2 - C_1| = C_2 - C_1 < d$, which implies $C_2 < C_1 + d$. The assumption that $C_1 < d$, leads to the contradiction $C_2 < 2d$ and $C_2 > 2d$. Hence, $\min\{C_1, C_2\} < d$ is impossible and the property holds. □

From Property 4.5 and (4), we can assume that for non-trivial instances of the problem:

$$d < p_{\text{sum}} < 3d. \tag{6}$$

5 Polynomial Time Approximation Scheme

We proposed an approximation scheme (cf., e.g., [39]), constructing a feasible solution for $P2|d_j = d|X$ based on a simplified instance of this problem. We adopted the idea of polynomial time approximation scheme given by Schuurman and Woeginger [40] for the two-machine problem with the makespan criterion, $P2||C_{\max}$, based on structuring problem input, which was originally proposed by Sahni [41].

Algorithm A_ε works in three phases. In Phase 1, for the original instance I of problem $P2|d_j = d|X$, with a set of jobs J, we construct the simplified instance $\bar{I}$, with a set of jobs $\bar{J}$, with regard to $0 < \varepsilon < 1$. We copy long jobs with $p_j > \varepsilon d$ of the total length L from I into $\bar{I}$, and replace short jobs with $p_j \leq \varepsilon d$ of the total length S in I by $\lfloor S/\varepsilon d \rfloor$ jobs with the processing time εd in $\bar{I}$. In the simplified instance $\bar{I}$, we have long jobs of the total length $\bar{L} = L$ and short jobs of the total length $\bar{S} = \lfloor S/\varepsilon d \rfloor \varepsilon d$. In Phase 2, we construct an optimal schedule for the simplified instance $\bar{I}$ with the optimal total early work $\bar{X}^*$. The workload $\bar{C_k}^*$ on machine M_k for $k = 1, 2$ contains $\bar{L_k}^*$ units of long jobs and $\bar{S_k}^*$ units of short jobs, i.e. $\bar{C_k}^* = \bar{L_k}^* + \bar{S_k}^*$, $\bar{L} = \bar{L_1}^* + \bar{L_2}^*$, $\bar{S} = \bar{S_1}^* + \bar{S_2}^*$. In Phase 3, presented without loss of generality for $\bar{C_1}^* \geq \bar{C_2}^*$ since the case for $\bar{C_1}^* < \bar{C_2}^*$ is solved analogously, we transform an optimal schedule for the simplified instance $\bar{I}$ into the feasible schedule for the original instance I with the total early work X. We assign in the schedule for I long jobs with $p_j > \varepsilon d$ to the same machine as in the optimal schedule for $\bar{I}$, at most $\bar{S_1}^* + 2\varepsilon d$ units of short jobs with $p_j \leq \varepsilon d$ to machine M_1, and the remaining jobs to machine M_2. The workload C_k on machine M_k for $k = 1, 2$ in the transformed schedule for instance I contains $L_k = \bar{L_k}^*$ units of long jobs and S_k units of short jobs, i.e. $C_k = L_k + S_k$.

In Theorem 5.1 we show that the above proposed method solves the considered problem in polynomial time. Then, in Theorem 5.2, we prove that it constructs schedules with guaranteed quality.

Theorem 5.1 *Algorithm A_ε is a polynomial algorithm for problem $P2|d_j = d|X$ with time complexity $O(\max\{n, \frac{1}{\varepsilon}2^{\frac{1}{\varepsilon}}\})$ for any given constant $0 < \varepsilon < 1$.*

Proof In Algorithm A_ε, simplifying instance I to instance $\bar{I}$ (Phase 1) and constructing the feasible solution for the original instance I based on the optimal schedule for $\bar{I}$ (Phase 3) takes $O(n)$ time. In Phase 2, we can use an enumeration algorithm to construct the optimal solution for the simplified instance $\bar{I}$, which explores all subsets of jobs, assigning them to M_1, and assigning the remaining jobs to M_2. Such an approach runs in $O(\frac{1}{\varepsilon}2^{\frac{1}{\varepsilon}})$ time, since the simplified instance $\bar{I}$ contains at most $\lfloor p_{\text{sum}}/\varepsilon d \rfloor$ jobs, and due to Property 4.5 and (6), we know that in non-trivial instances of the problem $p_{\text{sum}} < 3d$, and $\lfloor p_{\text{sum}}/\varepsilon d \rfloor < \lfloor 3d/\varepsilon d \rfloor < \lfloor 3/\varepsilon \rfloor$. Thus, the number of jobs in the simplified instance $\bar{I}$ is bounded by the constant $3/\varepsilon$, and it is independent of the original problem size. The enumeration method, checking all $2^{\frac{3}{\varepsilon}}$ subsets of jobs, each in time $O(3/\varepsilon)$ necessary to determine the criterion value, is a $O(\frac{1}{\varepsilon}2^{\frac{1}{\varepsilon}})$-time algorithm, which is polynomial for the original instance I (although it is obviously exponential with regard to $1/\varepsilon$). Thus, the overall time complexity of the method (Phases 1–3) is bounded by $O(\max\{n, \frac{1}{\varepsilon}2^{\frac{1}{\varepsilon}}\})$ for any given constant $0 < \varepsilon < 1$. □

To prove, in Theorem 5.2, that Algorithm A_ε is an approximation algorithm for the considered problem, we will compare in Lemma 5.1 optimal schedules for the original and simplified instances.

Lemma 5.1 *The optimal early work $\bar{X}^*$ for the simplified instance $\bar{I}$ is at least $(1-\varepsilon)$ times the optimal early work X^* for the original instance I, i.e. $\bar{X}^* > (1-\varepsilon)X^*$.*

Algorithm 1 Algorithm A_ε

Input: set of jobs $J = \{J_1, ..., J_n\}$ sorted in the non-increasing order of processing times p_j; set of machines $M = \{M_1, M_2\}$; common due date d; constant ε;

Phase 1: constructing a simplified instance $\bar{I}$

$J^L \leftarrow \{J_j \in J : p_j > \varepsilon d\}$;
$J^S \leftarrow \{J_j \in J : p_j \leq \varepsilon d\}$;
$\bar{J}^L \leftarrow \emptyset$;
for $j \leftarrow 1$ to $\left|J^L\right|$ **do**
    construct job $\bar{J}_j$ with $\bar{p}_j = p_j$;
    $\bar{J}^L \leftarrow \bar{J}^L \cup \{J_j\}$;
$\bar{J}^S \leftarrow \emptyset$;
$S = \sum_{j=|J^L|+1}^{|J|} p_j$;
for $j \leftarrow \left|J^L\right| + 1$ to $\left|J^L\right| + \lfloor S/\varepsilon d \rfloor$ **do**
    construct job $\bar{J}_j$ with $\bar{p}_j = \varepsilon d$;
    $\bar{J}^S \leftarrow \bar{J}^S \cup \{J_j\}$;
$\bar{J} \leftarrow \bar{J}^L \cup \bar{J}^S$;

Phase 2: solving the simplified instance $\bar{I}$

construct an optimal assignment of each job $\bar{J}_j \in \bar{J}$ to machine $M(\bar{J}_j) \in \{M_1, M_2\}$;
$\bar{C_1}^* = \sum_{\bar{J}_j \in \bar{J}: M(\bar{J}_j) = M_1} \bar{p}_j$;
$\bar{C_2}^* = \sum_{\bar{J}_j \in \bar{J}: M(\bar{J}_j) = M_2} \bar{p}_j$;

Step 3: constructing a feasible schedule for the original instance I, if $\bar{C_1}^* \geq \bar{C_2}^*$

for $j \leftarrow 1$ to $\left|J^L\right|$ **do**
    $M(J_j) \leftarrow M(\bar{J}_j)$;
$\bar{S_1}^* = \sum_{\bar{J}_j \in \bar{J}^S: M(\bar{J}_j) = M_1} \bar{p}_j$;
$k \leftarrow \left|J^L\right| + 1$;
$p \leftarrow p_k$;
while $(p < \bar{S_1}^* + 2\varepsilon d)$ **and** $(k \leq |J|)$ **do**
    $M(J_k) \leftarrow M_1$;
    $k \leftarrow k + 1$;
    $p \leftarrow p + p_k$;
for $j \leftarrow k$ **to** $|J|$ **do**
    $M(J_j) \leftarrow M_2$;
return.

Proof Let's consider the optimal schedule for the original instance I of the problem with the optimal early work $X^* = \min\{d, C_1^*\} + \min\{d, C_2^*\}$, where the workload on machine M_k is the sum of the length of long and short jobs, i.e. $C_k^* = L_k^* + S_k^*$ for $k = 1, 2$. We construct the feasible schedule for the simplified instance $\bar{I}$ by keeping on machine M_k the assignment of long jobs ($\bar{L_k} = L_k^*$) and replacing short jobs with $\left\lfloor S_k^*/\varepsilon d \right\rfloor$ jobs of length εd. In this schedule for instance $\bar{I}$ the early work equals $\bar{X} = \min\{d, \bar{C_1}\} + \min\{d, \bar{C_2}\}$ and $\bar{C_k} = \bar{L_k} + \bar{S_k} = L_k^* + \bar{S_k}$, where $\bar{S_k} = \left\lfloor \bar{S_k}^*/\varepsilon d \right\rfloor \varepsilon d$. According to Phase 1 of Algorithm A_ε we should have in the simplified instance $\bar{I}$ $\lfloor S/\varepsilon d \rfloor = \left\lfloor (S_1^* + S_2^*)/\varepsilon d \right\rfloor$ short jobs of length εd. Note that:

$$S_1^* + S_2^* = \left\lfloor (S_1^* + S_2^*)/\varepsilon d \right\rfloor \varepsilon d + r, 0 \leq r < \varepsilon d,$$
$$S_1^* = \left\lfloor S_1^*/\varepsilon d \right\rfloor \varepsilon d + r_1, 0 \leq r_1 < \varepsilon d,$$

$$S_2^* = \lfloor S_2^*/\varepsilon d \rfloor \varepsilon d + r_2, 0 \le r_2 < \varepsilon d,$$
$$\lfloor (S_1^* + S_2^*)/\varepsilon d \rfloor \varepsilon d + r = \lfloor S_1^*/\varepsilon d \rfloor \varepsilon d + r_1 + \lfloor S_2^*/\varepsilon d \rfloor \varepsilon d + r_2,$$
$$\lfloor (S_1^* + S_2^*)/\varepsilon d \rfloor \varepsilon d + r = (\lfloor S_1^*/\varepsilon d \rfloor + \lfloor S_2^*/\varepsilon d \rfloor)\varepsilon d + (r_1 + r_2).$$

Obviously, there is $0 \le (r_1 + r_2) < 2\varepsilon d$. If there is $0 \le (r_1 + r_2) < \varepsilon d$, then we have $\lfloor (S_1^* + S_2^*)/\varepsilon d \rfloor = \lfloor S_1^*/\varepsilon d \rfloor + \lfloor S_2^*/\varepsilon d \rfloor$, and the obtained schedule contains the expected number of short jobs, and it is a feasible schedule for the simplified instance $\bar{I}$. If $\varepsilon d \le (r_1 + r_2) < 2\varepsilon d$, then $\lfloor (S_1^* + S_2^*)/\varepsilon d \rfloor = \lfloor S_1^*/\varepsilon d \rfloor + \lfloor S_2^*/\varepsilon d \rfloor + 1$, and the obtained schedule has to be completed with one additional short job to be a feasible schedule for instance $\bar{I}$.

Case 1 $0 \le (r_1 + r_2) < \varepsilon d$

Because $\bar{S}_k = \lfloor S_k^*/\varepsilon d \rfloor \varepsilon d$ and $S_k^* = \lfloor S_k^*/\varepsilon d \rfloor \varepsilon d + r_k$, there is $S_k^* = \bar{S}_k + r_k$, $\bar{S}_k = S_k^* - r_k$ for $k = 1, 2$, and $\bar{C}_k = L_k^* + \bar{S}_k = L_k^* + S_k^* - r_k = C_k^* - r_k$. The difference in workloads on machine M_k in both schedules is equal to $C_k^* - \bar{C}_k = r_k$. Since there is $0 \le r_k < \varepsilon d$, the workloads on both machines can only decrease after the transformation of the schedule ($C_k^* - \bar{C}_k \ge 0$). The total early work may also decrease at most by $(C_1^* - \bar{C}_1) + (C_2^* - \bar{C}_2) = r_1 + r_2$ (note that decrease of workloads after the common due date does not influence the early work value, so the early work decrease might be smaller than the workloads decrease). Taking into account the assumption for Case 1 that $0 \le (r_1 + r_2) < \varepsilon d$, there is $0 \le (C_1^* - \bar{C}_1) + (C_2^* - \bar{C}_2) < \varepsilon d$, we have $\bar{X} \ge X^* - \{(C_1^* - \bar{C}_1) + (C_2^* - \bar{C}_2)\} > X^* - \varepsilon d$. Based on (5) we know that $X^* \ge d$ and $\bar{X} > X^* - \varepsilon d \ge X^* - \varepsilon X^* = (1 - \varepsilon)X^*$. Finally, since the optimal early work $\bar{X}^*$ for the simplified instance $\bar{I}$ may be only bigger than the early work $\bar{X}$ for the considered feasible schedule for this instance, $\bar{X}^* \ge \bar{X}$, we have $\bar{X}^* > (1 - \varepsilon)X^*$.

Case 2 $\varepsilon d \le (r_1 + r_2) < 2\varepsilon d$

To have the expected number of short jobs within the simplified instance $\bar{I}$, we complete the obtained schedule with an additional job of the length εd, which may be assigned to any machine. Let's assume without loss of generality that this additional job is assigned to machine M_1. Thus, the modified workloads on M_1 and M_2 in the schedule for $\bar{I}$ and differences in workloads values for instances I and $\bar{I}$ are as follows:

$$\bar{C}_1 = L_1^* + \bar{S}_1 = L_1^* + S_1^* - r_1 + \varepsilon d = C_1^* - r_1 + \varepsilon d \text{ and } C_1^* - \bar{C}_1 = r_1 - \varepsilon d,$$
$$\bar{C}_2 = L_2^* + \bar{S}_2 = L_2^* + S_2^* - r_2 = C_2^* - r_2 \text{ and } C_2^* - \bar{C}_2 = r_2.$$

Again the early work change is bounded by the workload changes on both machines, i.e. by $(C_1^* - \bar{C}_1) + (C_2^* - \bar{C}_2) = (r_1 + r_2) - \varepsilon d$. Taking into account the assumption for Case 2 that $\varepsilon d \le (r_1 + r_2) < 2\varepsilon d$, we have $0 \le (C_1^* - \bar{C}_1) + (C_2^* - \bar{C}_2) < \varepsilon d$ and $\bar{X} \ge X^* - \{(C_1^* - \bar{C}_1) + (C_2^* - \bar{C}_2)\} > X^* - \varepsilon d$. Again because $X^* \ge d$ and $\bar{X}^* \ge \bar{X}$, we conclude that $\bar{X}^* > (1 - \varepsilon)X^*$. □

Now we will estimate the quality of the schedule constructed by Algorithm A_ε in Phase 3, where the method transforms the optimal schedule for the simplified instance $\bar{I}$ into the feasible schedule for the original instance I.

Theorem 5.2 *The early work X for the feasible schedule determined by Algorithm A_ε for problem $P2|d_j = d|X$ is at least $(1 - 3\varepsilon)$ times the optimal early work X^*, i.e. $X > (1 - 3\varepsilon)X^*$.*

Proof Because Algorithm A_ε reserves on machine M_1 at most $\bar{S_1}^* + 2\varepsilon d$ units for short jobs from the original instance I, we have $S_1 \leq \bar{S_1}^* + 2\varepsilon d$. Moreover, since all short jobs have $p_j \leq \varepsilon d$, the unused interval on machine M_1, is strictly smaller than εd, i.e. $(\bar{S_1}^* + 2\varepsilon d) - S_1 < \varepsilon d$ (otherwise additional job would have been assigned by Algorithm A_ε to this machine). Hence, we have

$$\bar{S_1}^* + \varepsilon d < S_1 \leq \bar{S_1}^* + 2\varepsilon d. \tag{7}$$

The remaining short jobs from instance I are assigned to machine M_2 within interval $\bar{S_2}^*$. The duration of these jobs equals $S_2 = S - S_1$. Taking into account (7) we have $S - \bar{S_1}^* - \varepsilon d > S_2 \geq S - \bar{S_1}^* - 2\varepsilon d$. Due to Phase 1 of A_ε there is $\bar{S} = \lfloor S/\varepsilon d \rfloor \varepsilon d$ and $S \geq \bar{S} > S - \varepsilon d$. Since there is $\bar{S} = \bar{S_1}^* + \bar{S_2}^*$, we have $S \geq (\bar{S_1}^* + \bar{S_2}^*) > S - \varepsilon d$, which implies $\bar{S_2}^* > S - \bar{S_1}^* - \varepsilon d$. This means that the time reserved for short jobs on machine M_2 ($\bar{S_2}^*$) is big enough to schedule within it short jobs of length S_2 ($\bar{S_2}^* > S - \bar{S_1}^* - \varepsilon d > S_2$, i.e. $\bar{S_2}^* > S_2$). Moreover, because $S \geq \bar{S_1}^* + \bar{S_2}^*$, we have $\bar{S_2}^* \leq S - \bar{S_1}^* = S_1 + S_2 - \bar{S_1}^*$. Taking into account the relation given in (7), there is $\bar{S_2}^* \leq (\bar{S_1}^* + 2\varepsilon d) + S_2 - \bar{S_1}^* = S_2 + 2\varepsilon d$ and $S_2 \geq \bar{S_2}^* - 2\varepsilon d$. Summing up, we have

$$\bar{S_2}^* - 2\varepsilon d \leq S_2 < \bar{S_2}^*. \tag{8}$$

The change of workloads on machine M_k for $k = 1, 2$ between the schedule for $\bar{I}$ and I is determined by the change of the duration of short jobs, because the assignment of long jobs is identical in both schedules. We have $C_k = L_k + S_k = \bar{L_k}^* + S_k$ and $\bar{C_k}^* = \bar{L_k}^* + \bar{S_k}^*$, which implies that $(C_k - \bar{C_k}^*) = (S_k - \bar{S_k}^*)$ for $k = 1, 2$. On machine M_1, due to (7), we have

$$\varepsilon d < (C_1 - \bar{C_1}^*) = (S_1 - \bar{S_1}^*) \leq 2\varepsilon d. \tag{9}$$

On machine M_2, due to (8), we have

$$-2\varepsilon d \leq (C_2 - \bar{C_2}^*) = (S_2 - \bar{S_2}^*) < 0. \tag{10}$$

The workload on M_1 increases, after the transformation of the optimal schedule for the simplified instance $\bar{I}$ to the feasible schedule for the original instance I, causing possible increase of the early work on M_1, while the workload on M_2 decreases, causing possible decrease of the early work on M_2. According to Phase 3 of Algorithm A_ε, we know that $\bar{C_1}^* \geq \bar{C_2}^*$. Due to (9) and (10) we have $C_1 > \bar{C_1}^*$ and $C_2 < \bar{C_2}^*$ and in the constructed schedule there is $C_1 > C_2$. We have to consider two cases.

Case 1 $\bar{C_1}^* \geq d$

If in the optimal schedule for the simplified instance $\bar{I}$ there is $\bar{C_1}^* \geq d$, then in the final schedule for the original instance I, we have $C_1 > d$. This means that the change in the workload on M_1 does not influence the early work because it concerns jobs executed after the common due date. Consequently the total early work may only decrease, $\bar{X}^* \geq X$, due to workload decrease on machine M_2 (if $C_2 \geq d$, then the criterion value X does not change with regard to $\bar{X}^*$, because the workload on

M_2 exceeds the due date in both schedules). The decrease in the total early work is bounded by $\bar{C_2}^* - C_2$, i.e. $\bar{X}^* - X \leq \bar{C_2}^* - C_2 \leq 2\varepsilon d$ and, finally, $X \geq \bar{X}^* - 2\varepsilon d$.
Case 2 $\bar{C_1}^* < d$

If in the optimal schedule for the simplified instance $\bar{I}$ there is $\bar{C_1}^* < d$, then we have $C_2 < d$. In both schedules, the workloads on M_2 are strictly smaller than d, and the workload decrease on M_2 ($C_2 < \bar{C_2}^*$) reduces the early work on this machine by $-2\varepsilon d \leq \Delta X_2 < 0$ according to (10). On machine M_1, the workload increases ($C_1 > \bar{C_1}^*$), possibly increasing the early work at most by $2\varepsilon d$ units according to (9). However, if $C_1 > d$, then the increase in the early work is smaller than $C_1 - \bar{C_1}^*$, because a part of short jobs from the original instance I assigned to this machine by the algorithm are executed after d, and their processing times do not contribute to the criterion value. The increase of early work is bounded by $d - \bar{C_1}^* > 0$, instead of $(C_1 - \bar{C_1}^*) > \varepsilon d$. Summing up, the workload increase on M_1 increases the early work on this machine by $0 < \Delta X_1 \leq 2\varepsilon d$. Finally, in result of the instance transformation, the total early work changes by $-2\varepsilon d < \Delta X_1 + \Delta X_2 < 2\varepsilon d$, and the total early work for the constructed feasible schedule for the original instance I cannot be smaller than the optimal early work for simplified instance $\bar{I}$ by more than $2\varepsilon d$, i.e. $X > \bar{X}^* - 2\varepsilon d$. In both cases, $X > \bar{X}^* - 2\varepsilon d$. Taking into account Lemma 5.1, i.e. $\bar{X}^* > (1-\varepsilon)X^*$, Property 4.4 and (5), i.e. $X^* \geq d$, we have $X > (1-3\varepsilon)X^*$. □

Based on Theorems 5.1 and 5.2, a family of Algorithms A_ε is a polynomial time approximation scheme (PTAS) for problem $P2|d_j = d|X$.

6 Conclusions

Within this paper, we returned to the studies on the late work minimization/early work maximization on parallel identical machines, which is a classical scheduling model, finding many practical applications. We collected results on early/late work scheduling problems, which have been published in the literature since the last survey paper on this subject appeared. We discussed formally the relationship between the late work and the early work. We pointed out that these two criteria are equivalent when the optimal solutions are considered, but they have different nature when the approximate solutions are taken into account. The total late work problem, $P2|d_j = d|Y$, is non-approximable, while the total early work problem, $P2|d_j = d|X$, is approximable, since for the first problem no polynomial-time approximation scheme exists (unless P = NP), while for the latter one we proposed PTAS.

The computational experiments reported in this paper, performed for a few list scheduling methods, showed that developing more advanced optimization algorithms for the analyzed problems, such as metaheuristic algorithms, cannot be considered as real challenge. The quality of list scheduling solutions is close to the optimal ones for the considered case. Only very small instances, with very specific due date values, are difficult to be solved. Proposing metaheuristics was and still is a very popular research direction not only in the field of late work scheduling (cf., e.g., [16,19,24,32,42–44]). The results presented in this paper showed

that simple strategies, such as list algorithms, should not be a priori rejected from the analysis by assuming their low efficiency.

On the contrary to metaheuristics, approximation algorithms are still worth to be looked for, because they give the guarantee of the solution quality, which might be crucial for some practical applications. The analysis of computational results inspired theoretical studies on problem $P2|d_j = d|X$, which resulted in proving a few dominance properties helpful in constructing such approximation methods. Within the paper, we proposed the polynomial time approximation scheme for the considered problem, constructing schedules with guaranteed $(1-3\varepsilon)$ quality in $O(\max\{n, \frac{1}{\varepsilon}2^{\frac{1}{\varepsilon}}\})$ time, for any given constant $0 < \varepsilon < 1$.

Acknowledgements The authors would like to thank Vitaly A. Strusevich for helpful and valuable suggestions which allowed improving the quality of the paper.

References

1. Blazewicz, J., Ecker, K., Pesch, E., Schmidt, G., Weglarz, J.: Handbook on Scheduling. From Theory to Applications. Springer, Berlin (2007)
2. Pinedo, M.: Scheduling: Theory. Algorithms and Systems. Springer, New York (2008)
3. Blazewicz, J.: Scheduling preemptible tasks on parallel processors with information loss. Tech. Sci. Inform. **3**(6), 415–420 (1984)
4. Potts, C.N., Van Wassenhove, L.N.: Single machine scheduling to minimize total late work. Oper. Res. **40**(3), 586–595 (1991)
5. Blazewicz, J., Pesch, E., Sterna, M., Werner, F.: Flow shop scheduling with late work criterion–choosing the best solution strategy. In: Lecture Notes in Computer Science, vol. 3285, pp. 68–75. Springer-Verlag, Berlin, Heidelberg (2004)
6. Sterna, M.: Late work minimization in a small flexible manufacturing system. Comput. Ind. Eng. **52**(2), 210–228 (2007)
7. Ren, J., Du, D., Xu, D.: The complexity of two supply chain scheduling problems. Inf. Process. Lett. **113**(17), 609–612 (2013)
8. Sterna, M.: A survey of scheduling problems with late work criteria. Omega Int. J. Manag. Sci. **39**(2), 20–129 (2011)
9. Chen, X., Sterna, M., Han, X., Blazewicz, J.: Scheduling on parallel identical machines with late work criterion: offline and online cases. J. Sched. **19**(6), 729–736 (2016)
10. Hariri, A.M.A., Potts, C.N., Van Wassenhove, L.N.: Single machine scheduling to minimize total late work. ORSA J. Comput. **7**(2), 232–242 (1995)
11. Kovalyov, M.Y., Potts, C.N., Van Wassenhove, L.N.: A fully polynomial approximation scheme for scheduling a single machine to minimize total weighted late work. Math. Oper. Res. **19**(1), 86–93 (1994)
12. Blazewicz, J., Pesch, E., Sterna, M., Werner, F.: Open shop scheduling problems with late work criteria. Discrete Appl. Math. **134**(1), 1–24 (2004)
13. Blazewicz, J., Pesch, E., Sterna, M., Werner, F.: The two-machine flow-shop problem with weighted late work criterion and common due date. Eur. J. Oper. Res. **165**(2), 408–415 (2005)
14. Blazewicz, J., Pesch, E., Sterna, M., Werner, F.: A comparison of solution procedures for two-machine flow shop scheduling with late work criterion. Comput. Ind. Eng. **49**(4), 611–624 (2005)
15. Blazewicz, J., Pesch, E., Sterna, M., Werner, F.: A note on two-machine job shop with weighted late work criterion. J. Sched. **10**(2), 87–95 (2007)

16. Lin, B.M.T., Lin, F.C., Lee, R.T.C.: Two-machine flowshop scheduling to minimize total late work. Eng. Optim. **38**(4), 501–509 (2006)
17. Sterna, M.: Dominance relations for two-machine flow-shop problem with late work criterion. Bull. Pol. Acad. Sci. Tech. **55**(1), 59–69 (2007)
18. Leung, J.Y.-T.: Minimizing total weighted error for imprecise computation tasks and related problems. In: Leung, J.Y.-T. (ed.) Handbook of Scheduling: Algorithms, Models, and Performance Analysis, pp. 34.1–34.16. CRC Press, Boca Raton (2004)
19. Wu, C.C., Chen, H.M., Cheng, S.R., Hsu, C.J., Wu, W.H.: Simulated annealing approach for the single-machine total late work scheduling problem with a position-based learning. In: Volume 2 of IEEE 18th International Conference on Industrial Engineering and Engineering Management—Changchun, pp. 839–843. doi:10.1109/ICIEEM.2011.6035289 (2011)
20. Wu, C.C., Yin, Y., Wu, W.H., Chen, H.M., Cheng, S.R.: Using a branch and bound and a genetic algorithm for a single machine total late work scheduling problem. Soft Comput. **20**(4), 1329–1339 (2016)
21. Chen, X., Chau, V., Xie, P., Sterna, M., Blazewicz, J.: Complexity of late work minimization in flow shop systems and particle swarm optimization algorithm for learning effect. Comput. Ind. Eng. **111**, 176–182 (2017)
22. Zhang, X.G., Wang, Y.: Two-agent scheduling problems on a single-machine to minimize the total weighted late work. J. Comb. Optim. **33**(3), 945–955 (2017)
23. Wang, D.J., Kang, C.C., Shiau, Y.R., Wu, C.C., Hsu, P.H.: A two-agent single machine scheduling problem with late work criteria. Soft Comput. (2015). doi:10.1007/s00500-015-1900-5
24. Piroozfard, H., Wong, K.Y.: Job shop scheduling problem with late work criterion. In: American Institute of Physics (AIP) Conference Proceeding 1660/050061—Malaysia—Penang (2015). doi:10.1063/1.4915694
25. Afzalirad, M., Rezaeian, J.: Design of high-performing hybrid meta-heuristics for unrelated parallel machine scheduling with machine eligibility and precedence constraints. Eng. Optim. **48**(4), 706–726 (2016)
26. Abasian, F., Ranjbar, M., Salari, M., Davari, M., Khatami, S.M.: Minimizing the total weighted late work in scheduling of identical parallel processors with communication delays. Appl. Math. Model. **38**(15–16), 3975–3986 (2014)
27. Al Zuwaini, M.K., Zeyad, A.A.: Using branch and bound method to minimize bi-criteria. J. Progress. Res. Math. **7**(1), 907–915 (2016)
28. Ranjbar, M., Hosseinabadi, S., Abasian, F.: Minimizing total weighted late work in the resource-constrained project scheduling problem. Appl. Math. Model. **37**(23), 9776–9785 (2013)
29. Blazewicz, J., Finke, G.: Minimizing mean weighted execution time loss on identical and uniform processors. Inf. Process. Lett. **24**(4), 259–263 (1987)
30. Potts, C.N., Van Wassenhove, L.N.: Approximation algorithms for scheduling a single machine to minimize total late work. Oper. Res. Lett. **11**(5), 261–266 (1992)
31. Yin, Y.Q., Xu, J.Y., Cheng, T.C.E., Wu, C.C., Wang, D.J.: Approximation schemes for single-machine scheduling with a fixed maintenance activity to minimize the total amount of late work. Nav. Res. Logist. **63**(2), 172–183 (2016)
32. Xu, Z., Zou, Y., Kong, X.: Metaheuristic algorithms for parallel identical machines scheduling problem with weighted late work criterion and common due date. SpringerPlus **4**(782), 1–13 (2015). doi:10.1186/s40064-015-1559-5
33. Ben-Yehoshua, Y., Mosheiov, G.: A single machine scheduling problem to minimize total early work. Comput. Oper. Res. **73**, 115–118 (2016)
34. Gawiejnowicz, St: Time-Dependent Scheduling. Springer, Berlin (2008)
35. Agnetis, A., Billaut, J-Ch., Gawiejnowicz, St, Pacciarelli, D., Soukhal, A.: Multiagent Scheduling: Models and Algorithms. Springer, Berlin (2014)
36. Blazewicz, J., Pesch, E., Sterna, M., Werner, F.: Total late work criteria for shop scheduling problems. In: Inderfurth, K., Schwoediauer, G., Domschke, W., Juhnke, F., Kleinschmidt, P., Waescher, G. (eds.) Operations Research Proceedings 1999, pp. 354–359. Springer, Berlin (2000)
37. Alon, N., Azar, Y., Woeginger, G.J., Yadid, T.: Approximation schemes for scheduling on parallel machines. J. Sched. **1**(1), 55–66 (1998)
38. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. WH Freeman and Company, New York (1979)

39. Shachnai, H., Tamir, T.: Polynomial-time approximation schemes. In: Gonzalez, T. (ed.) Handbook of Approximation Algorithms and Metaheuristics, Chapter 9. Chapmann & Hall/CRC, Boca Raton (2007)
40. Schuurman, P., Woeginger, G.J.: Approximation schemes-A tutorial. In: Moehring, R.H., Potts, C.N., Schulz, A.S., Woeginger, G.J., Wolsey, L.A. (eds.): Lectures on Scheduling, http://www.win.tue.nl/~gwoegi/papers/ptas.pdf (2011). Accessed 06 Dec 2016
41. Sahni, S.: Approximate algorithms for the 0/1 knapsack problem. J. ACM **22**(1), 115–124 (1975)
42. Blazewicz, J., Pesch, E., Sterna, M., Werner, F.: Metaheuristics for late work minimization in two-machine flow shop with common due date. In: Lecture Notes Artificial Intelligence, vol. 3698, pp. 222–234. Springer-Verlag, Berlin, Heidelberg (2005)
43. Blazewicz, J., Pesch, E., Sterna, M., Werner, F.: Metaheuristic approaches for the two-machine flow-shop problem with weighted late work criterion and common due date. Comput. Oper. Res. **35**(2), 574–599 (2008)
44. Pesch, E., Sterna, M.: Late work minimization in flow shop by a genetic algorithm. Comput. Ind. Eng. **57**(4), 1202–1209 (2009)