Ann Oper Res (2012) 196:491–516
DOI 10.1007/s10479-012-1098-1

Scheduling problems with position dependent job processing times: computational complexity results

Radosław Rudek

Published online: 29 February 2012

Abstract In this paper, we analyse single machine scheduling problems with learning and aging effects to minimize one of the following objectives: the makespan with release dates, the maximum lateness and the number of late jobs. The phenomena of learning and aging are modeled by job processing times described by non-increasing (learning) or non-decreasing (aging) functions dependent on the number of previously processed jobs, i.e., a job position in a sequence. We prove that the considered problems are strongly NP-hard even if job processing times are described by simple linear functions dependent on a number of processed jobs. Additionally, we show a property of equivalence between problems with learning and aging models. We also prove that if the function describing decrease/increase of a job processing time is the same for each job then the problems with the considered objectives are polynomially solvable even if the function is arbitrary. Therefore, we determine the boundary between polynomially solvable and strongly NP-hard cases.

Keywords Scheduling · Learning effect · Aging effect · Computational complexity

1 Introduction

Changeability is a characteristic feature of many real-life systems (e.g., manufacturing, industrial, computer, etc.) that in general can be classified as an improvement or a degradation of such a system. The improvement was for the first time discovered and described in a quantitative form in aircraft industry by Wright (1936). He observed that the total hours to assemble an aircraft decreases as the number of assembled aircrafts increases due to the increasing experience of workers (the *learning effect*). Therefore, the same resources allow to produce more units in a shorter period of time. On this basis, Wright (1936) formulated a relation, called learning curve, where the time $p(v)$ required to produce the vth unit was defined as follows:

R. Rudek (✉)
Wrocław University of Economics, Komandorska 118/120, 53-345 Wrocław, Poland
e-mail: radoslaw.rudek@ue.wroc.pl

$$p(v) = av^{\alpha}, \tag{1}$$

where a is the time required to produce the first unit and $\alpha \leq 0$ is the learning index.

It is not surprising that the learning effect has attracted particular attention in the aircraft industry earlier than in any other on account of the high cost of aircrafts (Kerzner 1998). The benefits following the theory of the learning effect were soon recognized by USA War Production Board and the methodology proposed by Wright was used to plan the production of airplanes for the World War II needs (see Roberts 1983). Further empirical studies on the learning effect carried out within the last 60 years horizon proved its significant impact on productivity in manufacturing systems specialized in Hi-Tech electronic equipment (Adler and Clark 1991), memory chips and circuit boards (Webb 1994), electronic guidance systems (Kerzner 1998) and in many others (e.g. Carlson and Rowe 1976; Cochran 1960; Holzer and Riahi-Belkaoui 1986; Jaber and Bonney 1999; Lien and Rasch 2001; Yelle 1979). Most of these investigations confirmed the high accuracy of (1), however, they also revealed that some systems are more precisely described by other characteristics (learning curves), e.g., S-shaped, Stanford-B or DeJong (see Holzer and Riahi-Belkaoui 1986; Jaber and Bonney 1999; Lien and Rasch 2001).

In general the learning effect takes place in typical human activity environments or in automatized manufacturing, where a human support for machines is needed during activities such as operating, controlling, setup, cleaning, maintaining, failure removal, etc. Although learning can cease with time, it is often aroused by such factors as new inexperienced employees, extension of an assortment, new machines, more refined equipment, software update or general changes of the production environment (Biskup 2008).

However, the learning effect is not limited to the areas dominated by human. For instance, highly automatized manufacturing systems may benefit on the fact that if a machine does the same job repetitively, then the knowledge from the previous iterations can be used to improve the performance of a system when the job is processed the next time. An example of such method is iterative learning control that compensates a repetitive error in a robot motion control (see Arimoto et al. 1984).

The learning effect also occurs in machine learning and artificial intelligence. For instance, reinforcement learning algorithms, that usually learn and operate on-line (see Whiteson and Stone 2004), improve their efficiency on the basis of interactions with an environments (learning-by-doing). Thus, the performances of the systems optimized by such algorithms improve in their succeeding iterations (e.g., Buşoniu et al. 2008; Janiak and Rudek 2011).

The described learning effect that is a result of repeating similar operations (learning-by-doing) is called an autonomous learning (e.g., Yelle 1979). The theory of the learning effect enables (based on (1)) for efficient estimation of the variable production time and/or cost caused by learning. Thus, it allows to improve lot-sizes, worker management, energy/resource consumption, etc. (e.g., Keachie and Fontana 1966; Kerzner 1998; Li and Cheng 1994; Webb 1994). Nevertheless, it is not possible to optimize time and/or cost objectives beyond reductions resulting from learning-by-doing (Biskup 2008). It follows from the innate nature of autonomous learning and from an assumption of identical products, therefore, control (management) abilities provided by the theory of the learning effect are significantly limited.

However, in many manufacturing systems jobs (e.g., products) are not identical, but similar and the time required to process each of them can differ. This rigorous constraint on identical jobs was relaxed by Biskup (1999). Based on (1), he assumed that the time $p_j(v)$ required to process job (e.g., to produce unit) j decreases as the number v of processed similar (not necessarily identical) jobs increases and this relation was described as follows:

$$p_j(v) = a_j v^{\alpha}, \tag{2}$$

where a_j is the time required to process job j if no learning exists (i.e., it is processed as the first one). On this basis, a new model was obtained that offers an additional control variable, i.e., sequence of processed jobs. Thus, it became possible to optimize production objectives such as the maximum completion time of jobs, the maximum lateness or the number of late jobs (e.g. Bachman and Janiak 2004; Cheng and Wang 2000; Cheng et al. 2008; Lee and Lai 2011; Lee et al. 2010; Wu et al. 2007; Wang and Wang 2011; Yang and Kuo 2009; Zhang et al. 2011), which were beyond control using the theory of the learning effect. Therefore, fundamental for this approach is that control decisions (schedule) do not influence learning (that anyway in many cases is impossible), but they allow to efficiently utilize learning abilities of the system to optimize given objectives. Thus, not surprisingly, this direction of research has attracted particular attention in the scheduling theory, especially in issues devoted to manufacturing systems (for survey see Biskup 2008).

On the other hand, degradation of a system can be caused by *deterioration/aging* of machines (understood as lathe machines, chemical cleaning baths, etc.) or fatigue of human workers that affects the production parameters such as time/cost required to produce a single unit (e.g. Dababneh et al. 2001; Eilon 1964; Mandich 2003; Stanford and Lister 2004). Similarly as for learning systems, the objectives of deteriorating systems can also be controlled (in a specified range) by a schedule of processed jobs. In the scheduling theory there are two approaches to model deterioration. Although both of them describe the dependency between job processing times and deteriorating factors, for each of them the deteriorating factor is represented by different parameters. Namely, the first approach, called *deteriorating effect*, assumes that the job processing times are non-decreasing functions of their starting times and it has been extensively studied in the last decade (see Cheng et al. 2004 and Gawiejnowicz 2008).

However, scheduling models consistent with this approach are not relevant to many real-life industrial problems. It is especially significant for environments, where deterioration does not take place (or is negligible) during idle times of machines (workers), e.g., caused by different release dates of jobs. Such inconveniences are absent in the second approach, called *aging/fatigue effect*, in which job processing times are described by non-decreasing functions dependent on the actual condition (fatigue) of machines affected by already processed jobs (e.g., Cheng et al. 2010; Janiak and Rudek 2010; Kuo and Yang 2008; Rudek and Rudek 2011; Yang et al. 2010). Similarity of jobs usually allows to assume that each of them has the same impact on the fatigue of a machine (e.g. Cheng et al. 2008; Gawiejnowicz 1996; Mosheiov 2001; Yang and Yang 2010). Therefore, we will focus on this approach, where the time $p_j(v)$ required to process job j increases together with the number v of processed jobs. This relation can be described by (2), where $\alpha \geq 0$ (see Mosheiov 2001).

In this paper, we analyse computational complexity of single machine scheduling problems with linear models of learning/aging and the following minimization objectives: the maximum lateness, the makespan with release dates and additionally the number of late jobs. The main theoretical result of this paper is to prove that the considered problems are strongly NP-hard with linear functions of job processing times. Although the maximum lateness minimization scheduling problems with position dependent job processing times are broadly discussed (e.g., Bachman and Janiak 2004; Cheng and Wang 2000; Cheng et al. 2008; Lee and Lai 2011), their computational complexity is not fully determined. Therefore, to complement these results and to make their analysis coherent, we will determine their computational complexity. Namely, we will prove that the problems even with the simplest possible (nontrivial) mathematical models of job processing times are strongly NP-hard. Moreover, it will be shown that if the models are simpler (then trivial) the related problems are polynomially solvable. Thus, we will determine the boundary between the polynomial solvability of the problems and their strong NP-hardness. Thereby, we will complement the results provided *inter alia* by Bachman and Janiak (2004) and Cheng and Wang (2000) and foremost complete the results concerning the maximum lateness minimization with position dependent job processing times. Since, any NP-hardness proof is more significant if it is done for the simplest possible problem as it is done in this paper.

On the other hand, the practical aspect of this research is to show that the maximum acceptable simplification of job processing time functions (by their linearization) does not lead to decreasing complexity of the considered problems. Therefore, it does not lead to decreasing effort required to obtain optimal control decisions but rather to decreasing their accuracy. Additionally, we also prove that the considered problems with an arbitrary functions describing decreasing/increasing of a job processing time are polynomially solvable if the functions are the same for each job.

The remainder of this paper is organized as follows. Section 2 contains problem formulation. Computational complexity of the considered problems is determined in Sect. 3 and some polynomially solvable cases are provided in Sect. 4. Finally Sect. 5 concludes the paper.

2 Problem formulation and notation

In this section, we will formulate scheduling problems with two phenomenon: aging (fatigue) and learning.

There is given a single machine and a set $J = \{1, \dots, n\}$ of n jobs (e.g., tasks, products, cleaned items) that have to be processed by a machine; there are no precedence constraints between jobs. The machine is continuously available and can process at most one job at a time. Once it begins processing a job it will continue until this job is finished. Each job is characterized by its aging/learning curve $p_j(v)$ that describes increasing/decreasing of the time required to process this job depending on the number of jobs completed before it. In other words, we will say that $p_j(v)$ is a processing time of job j if it is processed as the vth job in a sequence. Moreover, each job j is also characterized by the normal processing time a_j that is the time required to process the job if the machine is not influenced by aging/learning (i.e., $a_j \triangleq p_j(1)$). Other job parameters are the release date r_j that is the time at which the job is available for processing and the due-date d_j when it should be completed.

For the aging effect, the processing time (aging/fatigue curve) of job j is described by a linear function of its position v in a sequence:

$$p_j(v) = a_j v. \tag{3}$$

On the other hand, for the learning effect the processing time (learning curve) is given as follows:

$$p_j(v) = a - b_j v, \tag{4}$$

where a is the normal processing time common for all jobs ($a_j = a$ for $i = 1, \dots, n$) and b_j is a learning ratio of job j. Thus, we consider the simplest linear aging/learning models for processing non-identical jobs (parameters are not common for all jobs).

We also consider problems where learning/aging curves are identical for all jobs. The processing times of such jobs are described as follows:

$$p_j(v) = a_j + f(v), \tag{5}$$

where $f(v)$ is an arbitrary function of a job position in a sequence, such that $f(1) = 0$ and $a_j + f(v) > 0$ for $j, v = 1, \ldots, n$. Note that $f(v)$ models both learning ($f(v) < 0$) and aging ($f(v) > 0$) for $v = 1, \ldots, n$.

As it was mentioned in the previous section, the objectives of the considered aging/learning systems can be controlled by the sequence (schedule) of processed jobs (e.g. manufactured products). Therefore, let us define control variables (schedule) formally.

Let $\pi = \langle \pi(1), \ldots, \pi(i), \ldots, \pi(n) \rangle$ denote the sequence of jobs (permutation of the elements of the set J), where $\pi(i)$ is the job processed in position i in this sequence. By Π we will denote the set of all such permutations. For the given sequence (permutation) π, we can easily determine the completion time $C_{\pi(i)}$ of a job placed in the ith position in π from the following recursive formulae:

$$C_{\pi(i)} = \max\{C_{\pi(i-1)}, r_{\pi(i)}\} + p_{\pi(i)}(i), \tag{6}$$

where $C_{\pi(0)} = 0$ and the lateness $L_{\pi(i)}$ is defined as follows:

$$L_{\pi(i)} = C_{\pi(i)} - d_{\pi(i)}. \tag{7}$$

We will say that job $\pi(i)$ is late if $L_{\pi(i)} > 0$. The objective is to find such an optimal control, i.e., sequence (schedule) $\pi^* \in \Pi$ of jobs on the single machine, which minimizes one of the following objective functions: the maximum completion time (makespan) $C_{\max} \triangleq \max_{i=1,\ldots,n}\{C_{\pi^*(i)}\}$ (i.e., $C_{\max} \triangleq C_{\pi^*(n)}$), the maximum lateness $L_{\max} \triangleq \max_{i=1,\ldots,n}\{L_{\pi^*(i)}\}$ and the number of late jobs $\sum_{i=1}^{n} U_{\pi^*(i)}$, where

$$U_{\pi^*(i)} = \begin{cases} 0, & C_{\pi^*(i)} \le d_{\pi^*(i)} \\ 1, & C_{\pi^*(i)} > d_{\pi^*(i)} \end{cases}$$

and $U_{\pi^*(i)} = 1$ means that job $\pi^*(i)$ is late.

Formally the optimal control (schedule) $\pi^* \in \Pi$ for the considered minimization objectives is defined as follows $\pi^* \triangleq \operatorname{argmin}_{\pi\in\Pi}\{C_{\pi(n)}\}$, $\pi^* \triangleq \operatorname{argmin}_{\pi\in\Pi}\{\max_{i=1,\ldots,n}\{L_{\pi(i)}\}\}$, and $\pi^* \triangleq \operatorname{argmin}_{\pi\in\Pi}\{\sum_{i=1}^{n} U_{\pi(i)}\}$, respectively.

For convenience and to keep an elegant description of the considered problems we will use the three field notation scheme $X|Y|Z$ (see Graham et al. 1979), where X describes the machine environment, Y describes job characteristics and constraints and Z represents the minimization objectives. According to this notation, the problems will be denoted as follows: $1|r_j, ALE|C_{\max}$, $1|ALE|L_{\max}$ and $1|ALE|\sum U_j$, where $ALE \in \{p_j(v) = a_j v,\ p_j(v) = a - b_j v,\ p_j(v) = a_j + f(v)\}$. If $r_j = 0$ for $j = 1, \ldots, n$, then it is omitted in the given notation.

3 Computational complexity

In this section, we will prove that the considered problems are strongly NP-hard. First, we will determine the computational complexity of the maximum lateness minimization problem with the aging effect and next with the learning effect. The strong NP-hardness proofs for both problems are similar and based on the same idea. However, the problem with aging is simpler, thus, it is analyzed in the first order. From the results for the maximum lateness minimization problems follows the strong NP-hardness of the minimization of the number of late jobs with aging/learning. Next we will prove, on the basis of a problem equivalency, the makespan minimization with release dates is also strongly NP-hard with aging/learning models.

3.1 Aging effect

At first note that the problem $1|p_j(v) = a'_j + b'_j v|L_{\max}$ (where a'_j is the normal processing time of job j and b'_j is its aging ratio) was proved to be strongly NP-hard (Bachman and Janiak 2004). However, we will show that the simplest (nontrivial) problem $1|p_j(v) = a_j v|L_{\max}$ is strongly NP-hard. To do it, we will provide the pseudopolynomial time transformation from the strongly NP-complete problem 3-Partition (Garey and Johnson 1979) to the decision version of the considered scheduling problem, $1|p_j(v) = a_j v|L_{\max}$.

3-Partition (3PP) (Garey and Johnson 1979) There are given positive integers m, B and $x_1, \dots, x_{3m}$ of $3m$ positive integers satisfying $\sum_{q=1}^{3m} x_q = mB$ and $\frac{B}{4} < x_q < \frac{B}{2}$ for $q = 1, \dots, 3m$. Does there exist a partition of the set $Y = \{1, \dots, 3m\}$ into m disjoint subsets $Y_1, \dots, Y_m$ such that $\sum_{q \in Y_i} x_q = B$ for $i = 1, \dots, m$?

The decision version of the problem $1|p_j(v) = a_j v|L_{\max}$ (DAEL) is given as follows: *Does there exist such a schedule π of jobs on the machine for which $L_{\max} \le y$?*

At first, we will present the main idea of the proof. There are given $3m$ *partition* jobs (constructed on the basis of the elements from the set Y of 3PP) and mN *enforcer* jobs (where $N = mB$). The instances of DAEL are constructed such that the optimal schedules have the following properties: the enforcer jobs are partitioned into m subsets $E_1, \dots, E_m$ such that each consists of N jobs, partition jobs are partitioned into m subsets $X_1, \dots, X_m$ such that each consists of exactly 3 jobs and the optimal schedule has the following form $(E_1, X_1, E_2, X_2, E_3, \dots, E_i, X_i, E_{i+1}, \dots, X_{m-1}, E_m, X_m)$, where jobs within each subset are scheduled arbitrary. If a schedule is not consistent with these properties, then the criterion value $L_{\max}$ is always greater than a given value y (i.e., it cannot be optimal). On this basis, we will show that the answer for the constructed instances of DAEL is *yes* (i.e., $L_{\max} \le y$) if and only if it is *yes* for 3PP (i.e., $\sum_{q \in Y_i} x_q = B$ for $i = 1, \dots, m$).

The formal transformation from 3PP to DAEL is given as follows. The instance of DAEL contains the set $X = \{1, \dots, 3m\}$ of $3m$ *partition* jobs (constructed on the basis of the elements from the set Y of 3PP) and the set $E = \{e_1, \dots, E_{mN}\}$ of mN *enforcer* jobs. The enforcer jobs can be partitioned into m sets $E_i = \{e_{N(i-1)+1}, \dots, e_{Ni}\}$ for $i = 1, \dots, m$, such that jobs within each set E_i have the same parameters, i.e., $a_k = a_l$ and $d_k = d_l$ for $k, l \in E_i$ for $i = 1, \dots, m$.

The parameters of the enforcer jobs are defined as follows:

$$a_{e_{N(i-1)+1}} = \dots = a_{e_{Ni}} = a_{E_i} = a_E = \frac{1}{m(N+3)},$$

$$d_{e_{N(i-1)+1}} = \dots = d_{e_{Ni}} = d_{E_i} = \sum_{l=1}^{i-1} (W_l + V_l) + W_i,$$

for $i = 1, \dots, m$ where

$$\begin{aligned}
N &= mB,\\
M &= (m+1)^2(N+3)B,\\
V_i &= 3M\big(i(N+3)-1\big) + i(N+3)B - \frac{3}{4}B, \qquad (8)\\
W_i &= a_E\left((i-1)N(N+3) + \sum_{l=1}^{N} l\right), \qquad (9)
\end{aligned}$$

for $i = 1, \dots, m$ and the parameters of the partition jobs are

$$\begin{aligned}
a_j &= (M + x_j),\\
d_j &= D = \sum_{i=1}^{m}(W_i + V_i),
\end{aligned}$$

for $j = 1, \dots, 3m$ and $y = 0$.

Observe that each parameter of DAEL can be calculated in a time bounded by the polynomial dependent on m and B. Moreover, the maximum value of DAEL does not increase exponentially in reference to 3PP (i.e., D is $O(m^6B^3)$) and the problem size does not decrease exponentially in reference to 3PP (i.e., $n = O(m^2B)$). Thus, the transformation from 3PP to DAEL is pseudopolynomial.

Let X_i denote the set of the partition jobs that are processed just after jobs from the set E_i (for $i = 1, \dots, m$). Define a schedule π^*, where jobs are scheduled as follows: $(E_1, X_1, E_2, X_2, E_3, \dots, E_i, X_i, E_{i+1}, \dots, X_{m-1}, E_m, X_m)$, where $X_i = \{3i-2, 3i-1, 3i\}$ for $i = 1, \dots, m$, if it is not a case we can always renumber the partition jobs. Let $V(X_i)$ and W_i denote the sum of processing times of the partition jobs from X_i and the enforcer jobs from E_i, respectively, for the schedule π^*. Based on the transformation $V(X_i)$ is defined as:

$$V(X_i) = 3M\big(i(N+3)-1\big) + i(N+3)\sum_{q\in X_i} x_q - 2x_{3i-2} - x_{3i-1},$$

for $i = 1, \dots, m$ and it can be estimated as follows:

$$\begin{aligned}
&3M\big(i(N+3)-1\big) + i(N+3)\sum_{q\in X_i} x_q - \frac{3}{2}B\\
&\quad < V(X_i) < 3M\big(i(N+3)-1\big) + i(N+3)\sum_{q\in X_i} x_q - \frac{3}{4}B. \qquad (10)
\end{aligned}$$

It is easy to observe that the sum of processing times of the enforcer jobs from the set E_i, i.e., W_i, $(i = 1, \dots, m)$ in schedule π^* is given by (9). The completion time of the last job in E_i is C_{E_i} and of the last job in X_i is C_{X_i} for $i = 1, \dots, m$.

Let us also define useful inequalities:

$$V(X_i) > V_i - i(N+3)B - \frac{3}{4}B, \qquad (11)$$

$$W_i < a_E N\left((m-1)(N+3) + \frac{N+1}{2}\right) < a_E mN(N+3) = N, \qquad (12)$$

$$M > m(m+1)(N+3)B + mB + N > \sum_{l=1}^{m}\left(l(N+3)B + \frac{3}{4}B\right) + W_i, \qquad (13)$$

for $i = 1, \dots, m$. Note also that the processing times of the partition jobs can be estimated as follows $p_j(v) > M$ for $j = 1, \dots, 3m$ and $v = 1, \dots, m(N+3)$.

On this basis, we will provide properties of an optimal solution for DAEL.

Lemma 1 *The optimal sequence of jobs for the problem* $1|p_j(v) = av, d_j = d|L_{\max}$ *is arbitrary.*

Proof Trivial. □

Lemma 2 *The problem* $1|p_j(v) = av|L_{\max}$ *can be solved in* $O(n \log n)$ *steps by scheduling jobs according to the non-decreasing order of their due dates* (*the EDD rule*).

Proof Trivial. □

Lemma 3 *The problem* $1|p_j(v) = a_j v|C_{\max}$ *can be solved in* $O(n \log n)$ *steps by scheduling jobs according to the non-increasing order of their normal processing times* (*LPT rule*).

Proof Trivial. □

Based on the above lemmas we will prove the following.

Lemma 4 *There is an optimal schedule* π, *for the given instance of DAEL*, *in which*, *before the enforcer jobs form* E_i $(i = 1, \dots, m)$ *at last* $3(i-1)$ *partition jobs can be scheduled.*

Proof See Appendix. □

Lemma 5 *Jobs in each block* E_i *are processed one after another and between* E_i *and* E_{i+1} *exactly* 3 *partition jobs are scheduled for* $i = 1, \dots, m-1$.

Proof See Appendix. □

Based on the above lemmas, we will prove the following theorem.

Theorem 1 *The problem* $1|p_j(v) = a_j v|L_{\max}$ *is strongly NP-hard.*

Proof Based on the given transformation from 3PP to DAEL and on Lemma 5 we construct a schedule π for DAEL, that is given as follows: $(E_1, X_1, E_2, X_2, E_3, \dots, E_i, X_i, E_{i+1}, \dots, X_{m-1}, E_m, X_m)$. Recall that blocks of the enforcer jobs are scheduled according to the EDD rule and the schedule of jobs within each E_i is immaterial and the sequence of jobs within each set X_i is arbitrary. To make the calculations easier, renumber the jobs in these sets, i.e., $X_i = \{3i-2, 3i-1, 3i\}$ for $i = 1, \dots, m$.

Now we will show that the answer for DAEL is *yes* (i.e., $L_{\max} \le y$) *if and only if* it is *yes* for 3PP (i.e., $\sum_{q \in Y_i} x_q = B$ for $i = 1, \dots, m$).

"*Only if*." Assume that the answer for 3PP is *yes*. Thus, for each subset Y_i $(i = 1, \dots, m)$ holds $\sum_{q \in Y_i} x_q = B$, thereby, for each X_i also holds $\sum_{q \in X_i} x_q = B$. Therefore, $V(X_i) < V_i$

for $i = 1, \ldots, m$. Obviously, $C_{E_1} = d_{E_1}$ for schedule π regardless of a solution of 3PP. The completion times of the enforcer jobs for the schedule π are as follows:

$$C_{E_2} = W_1 + V(X_1) + W_2 < W_1 + V_1 + W_2 = d_{E_2},$$

$$C_{E_3} = \sum_{l=1}^{2}\big(W_l + V(X_l)\big) + W_3 < \sum_{l=1}^{2}(W_l + V_l) + W_3 = d_{E_3},$$

$$C_{E_i} = \sum_{l=1}^{i-1}\big(W_l + V(X_l)\big) + W_i < \sum_{l=1}^{i-1}(W_l + V_l) + W_i = d_{E_i}.$$

Thus, $C_{E_i} < d_{E_i}$ for $i = 1, \ldots, m$ and

$$C_{X_m} = C_{E_m} + V(X_m) < C_{E_m} + V_m < \sum_{i=1}^{m}(W_i + V_i) = D.$$

Thus, $L_{\max}(\pi) \le y = 0$, thereby DAEL has the answer *yes*.

"*If.*" Assume now that the answer for 3PP is *no*. Therefore, there is no partition of the set Y such that $\sum_{q\in Y_i} x_q = B$ holds for all $i = 1, \ldots, m$, thereby $\sum_{q\in X_i} x_q = B$ does not hold for $i = 1, \ldots, m$. Note that $|X_i| = 3$ for $i = 1, \ldots, m$ (follows from Lemma 5) regardless of the partition of 3PP.

Let $\sum_{q\in X_i} x_q = B + \lambda_i$ for $i = 1, \ldots, m$ and from the assumption of 3PP $\frac{B}{4} < x_q < \frac{B}{2}$ (for $q = 1, \ldots, 3m$) follows that $\frac{3}{4}B < \sum_{q\in X_i} x_q < \frac{3}{2}B$, thereby $\lambda_i \in (-\frac{B}{4}, \frac{B}{2})$.

Thus, for any partition of the set $\{1, \ldots, 3m\}$ into disjoint subsets $X_1, \ldots, X_m$, there must exist at least two subsets X_u and X_w ($u \ne w$) such that $\sum_{q\in X_u} x_q \ne \sum_{q\in X_w} x_q$ for $u, w \in \{1, \ldots, m\}$ and $u < w$. For this proof, it is sufficient to consider only two cases, since any distribution of λ_i (following the partition of jobs) can be represented by these cases. They are given as follows:

(a) $\lambda_u > 0$ and $\lambda_w < 0$, such that $\sum_{i=1}^{u-1} \lambda_i = 0$ and w is the index of the first set X_w for which $\sum_{l=u}^{w} \lambda_l \le 0$, i.e., $\sum_{l=u}^{i} \lambda_l > 0$ for $i = u, \ldots, w-1$,
(b) $\lambda_u < 0$ and $\lambda_w > 0$, such that $\sum_{i=1}^{u-1} \lambda_i = 0$ and w is the index of the first set X_w for which $\sum_{l=u}^{w} \lambda_l \ge 0$, i.e., $\sum_{l=u}^{i} \lambda_l < 0$ for $i = u, \ldots, w-1$,

where $u, w \in \{1, \ldots, m\}$ and $u < w$. Consider case (a) and assume that X_w is the first one such that $\sum_{i=u}^{w} \lambda_i \le 0$ and if $\sum_{i=u}^{w} \lambda_i + \lambda_{w+1} < 0$ (i.e., $\lambda_{w+1} < 0$), then there must exist such $k > w+1$, for which $\lambda_k > 0$ and $\sum_{i=w+1}^{k} \lambda_i \ge 0$, but this is represented by case (b). Thus, without loss of generality, we assume that $\lambda_i = 0$ for $i \in \{1, \ldots, u-1\} \cup \{w+1, \ldots, m\}$.

Based on (8) and (10) for $i = 1, \ldots, u-1$ ($\lambda_i = 0$) we have $V(X_i) > V_i - \frac{3}{4}B$. Following this, we can estimate the completion time of the last job in job E_{u+1}:

$$\begin{aligned}C_{E_{u+1}} &= \sum_{l=1}^{u-1}\big(W_l + V(X_l)\big) + W_u + V(X_u) + W_{u+1}\\ &> \sum_{l=1}^{u-1}\left(W_l + V_l - \frac{3}{4}B\right) + W_u + W_{u+1}\\ &\quad + 3M\big(u(N+3) - 1\big) + u(N+3)(B + \lambda_u) - \frac{3}{2}B\end{aligned}$$

$$= \sum_{l=1}^{u}(W_l + V_l) + W_{u+1} + u(N+3)\lambda_u - \frac{3}{4}Bu$$
$$= d_{E_{u+1}} + (N+3)u\lambda_u - \frac{3}{4}Bu.$$

Since $\lambda_u \in [1, \frac{B}{2})$ and $N = mB > \frac{3}{4}B$, then $C_{E_{u+1}} > d_{E_{u+1}}$, thereby $L_{\max} > y = 0$.

Consider now case (b). The completion time of the last job in E_{u+1} can be estimated as follows:

$$C_{E_{u+1}} > d_{E_{u+1}} + (N+3)u\lambda_u - \frac{3}{4}Bu.$$

Following this way the completion time of the last job in E_i for $i = u+1, \ldots, w+1$ (and $w \le m-1$) can be estimated:

$$C_{E_i} > d_{E_i} + (N+3)\sum_{l=u}^{i-1} l\lambda_l - \frac{3}{4}B(i-1).$$

On this basis and taking into consideration $\sum_{i=u}^{w} i\lambda_i = w\sum_{i=u}^{w}\lambda_i - \sum_{i=u}^{w-1}\sum_{l=u}^{i}\lambda_l$, the completion time of the last job in E_{w+1} (where $w \le m-1$) can be estimated:

$$C_{E_{w+1}} > d_{E_{w+1}} + (N+3)\sum_{i=u}^{w} i\lambda_i - \frac{3}{4}Bw$$
$$= d_{E_{w+1}} + (N+3)\left(w\sum_{i=u}^{w}\lambda_i - \sum_{i=u}^{w-1}\sum_{l=u}^{i}\lambda_l\right) - \frac{3}{4}Bw.$$

Since $\sum_{i=u}^{w}\lambda_i = 0$ and $\sum_{l=u}^{i}\lambda_l < 0$ for $u \le i < w$, then $\sum_{i=u}^{w-1}\sum_{l=u}^{i}\lambda_l < 0$, thereby

$$C_{E_{w+1}} > d_{E_{w+1}} + (N+3) - \frac{3}{4}Bw > d_{E_{w+1}},$$

for $w \le m-1$. If $w = m$, then the completion time of the last scheduled job in X_m can be estimated as follows:

$$C_{X_m} > \sum_{i=1}^{m}(W_i + V_i) + (N+3)\sum_{i=u}^{m} i\lambda_i - \frac{3}{4}Bm$$
$$= D + (N+3)\left(w\sum_{i=u}^{m}\lambda_i - \sum_{i=u}^{m-1}\sum_{l=u}^{i}\lambda_l\right) - \frac{3}{4}Bm$$
$$> D + (N+3) - \frac{3}{4}Bm > D.$$

Therefore, for all the cases the criterion value $L_{\max}(\pi)$ is greater than y.

We hereby showed that DAEL has an answer *yes* if and only if the answer for 3PP is also *yes*, which means DAEL is strongly NP-complete, thereby the considered scheduling problem $1|p_j(v) = a_j v|L_{\max}$ is strongly NP-hard. □

Note that any further relaxation of the problem $1|p_j(v) = a_j v|L_{\max}$ is polynomially solvable, namely $1|p_j(v) = av|L_{\max}$ (the EDD rule, see Lemma 2) and $1|p_j(v) = a_j v, d_j =$

$d|L_{\max}$ that is equivalent to $1|p_j(v)=a_jv|C_{\max}$ (see Lemma 3). Therefore, we also determine the boundary between polynomially solvable and NP-hard cases.

Since $1|p_j(v)=a_jv|L_{\max}$ is strongly NP-hard, thereby $1|p_j(v)=a_jv|\sum U_j$ is not less complex.

3.2 Learning effect

Cheng and Wang (2000) proved that the problem $1|p_j(v)=a_j-b_j\min\{v-1,g_j\}|L_{\max}$ is strongly NP-hard. However, we will show that even the significantly simpler problem $1|p_j(v)=a-b_jv|L_{\max}$ (with linear job processing times) is strongly NP-hard. Thus, we decrease the boundary between polynomially solvable and NP-hard cases of the maximum lateness minimization problems with position dependent job processing times.

The strong NP-hardness of $1|p_j(v)=a-b_jv|L_{\max}$ will be proved in the similar manner as in the case of the problem $1|p_j(v)=a_jv|L_{\max}$ and the main idea of the proof is exactly the same.

At first, we will provide the pseudopolynomial time transformation from the strongly NP-complete problem 3-PARTITION (Garey and Johnson 1979) to the decision version of the considered scheduling problem, $1|p_j(v)=a-b_jv|L_{\max}$.

The decision version of the problem $1|p_j(v)=a-b_jv|L_{\max}$ (DLEL) is given as follows: *Does there exist such a schedule π of jobs on the machine for which $L_{\max}\le y$?*

The pseudopolynomial time transformation from 3PP to DLEL is given. The constructed instance of DLEL contains the set $X=\{1,\dots,3m\}$ of $3m$ *partition* jobs (constructed on the basis of the elements from the set Y of 3PP) and the set $E=\{e_1,\dots,E_{mN}\}$ of mN *enforcer* jobs, where $N=mB$. The enforcer jobs can be partitioned into m sets $E_i=\{e_{N(i-1)+1},\dots,e_{Ni}\}$ for $i=1,\dots,m$, such that jobs within each set E_i have the same parameters, i.e., $b_k=b_l$ and $d_k=d_l$ for $k,l\in E_i$ for $i=1,\dots,m$.

Similarly as in the proof of Theorem 1, the parameters of the enforcer jobs are defined as follows:

$$a_{e_{N(i-1)+1}}=\cdots=a_{e_{Ni}}=a_{E_i}=a=2mM(N+3)+2mN(N+3)b_E,$$
$$b_{e_{N(i-1)+1}}=\cdots=b_{e_{Ni}}=b_{E_i}=b_E=M(N+1),$$
$$d_{e_{N(i-1)+1}}=\cdots=d_{e_{Ni}}=d_{E_i}=\sum_{l=1}^{i-1}(W_l+V_l)+W_i,$$

for $i=1,\dots,m$, where

$$N=mB,$$
$$M=(m+1)^2(N+3)B,$$
$$V_i=3a-3M\big(i(N+3)-1\big)+i(N+3)B-\frac{3}{4}B, \tag{14}$$
$$W_i=aN-b_E\left((i-1)N(N+3)+\sum_{l=1}^{N}l\right), \tag{15}$$

for $i=1,\dots,m$ and of the partition jobs

$$a_j=a,$$

$$b_j = (M - x_j),$$

$$d_j = D = \sum_{i=1}^{m}(W_i + V_i),$$

for $j = 1, \ldots, 3m$ and $y = 0$.

Observe that each parameter of DLEL can be calculated in a time bounded by the polynomial dependent on m and B. Moreover, the maximum value of DLEL does not increase exponentially in reference to 3PP (i.e., D is $O(m^9B^6)$) and the problem size does not decrease exponentially in reference to 3PP (i.e., $n = O(m^2B)$). Thus, the transformation from 3PP to DLEL is pseudopolynomial.

Let X_i denote the set of the partition jobs that are processed just after jobs from the set E_i (for $i = 1, \ldots, m$). Define a schedule π^*, where jobs are scheduled as follows: $(E_1, X_1, E_2, X_2, E_3, \ldots, E_i, X_i, E_{i+1}, \ldots, X_{m-1}, E_m, X_m)$, where $X_i = \{3i-2, 3i-1, 3i\}$ for $i = 1, \ldots, m$, if it is not a case we can always renumber the partition jobs. Let $V(X_i)$ and W_i denote the sum of processing times of the partition jobs from X_i and the enforcer jobs from E_i, respectively, for the schedule π^*. Based on the transformation $V(X_i)$ is defined as:

$$V(X_i) = 3a - 3M\big(i(N+3) - 1\big) + i(N+3)\sum_{q\in X_i} x_q - 2x_{3i-2} - x_{3i-1},$$

for $i = 1, \ldots, m$ and it can be estimated as follows:

$$\begin{aligned}&3a - 3M\big(i(N+3) - 1\big) + i(N+3)\sum_{q\in X_i} x_q - \frac{3}{2}B\\ &\quad < V(X_i) < 3a - 3M\big(i(N+3) - 1\big) + i(N+3)\sum_{q\in X_i} x_q - \frac{3}{4}B.\end{aligned} \tag{16}$$

It is easy to observe that the sum of processing times of the enforcer jobs from the set E_i, i.e., W_i ($i = 1, \ldots, m$) in schedule π^* is given by (15). The completion time of the last job in E_i is C_{E_i} and of the last job in X_i is C_{X_i} for $i = 1, \ldots, m$.

Let us also define useful inequalities:

$$V(X_i) > V_i - i(N+3)B - \frac{3}{4}B, \tag{17}$$

$$M > \sum_{l=1}^{m}\left(l(N+3)B + \frac{3}{4}B\right), \tag{18}$$

$$a > b_E mN(N+3) + MmN(N+3) + \sum_{l=1}^{m}\left(l(N+3)B + \frac{3}{4}B\right), \tag{19}$$

for $i = 1, \ldots, m$.

On this basis, we will provide properties of an optimal solution for DLEL.

Lemma 6 *The optimal sequence of jobs for the problem* $1|p_j(v) = a - bv, d_j = d|L_{\max}$ *is arbitrary.*

Proof Trivial. □

Lemma 7 *The problem $1|p_j(v) = a - bv|L_{\max}$ can be solved in $O(n \log n)$ steps by scheduling jobs according to the non-decreasing order of their due dates (the EDD rule).*

Proof Trivial. □

Lemma 8 *The problem $1|p_j(v) = a - b_j v|C_{\max}$ can be solved in $O(n \log n)$ steps by scheduling jobs according to the non-decreasing order of b_j parameters.*

Proof Trivial. □

Based on the above lemmas we will prove the following.

Lemma 9 *There is an optimal schedule π, for the given instance of DLEL, in which, before the enforcer jobs from E_i $(i = 1, \dots, m)$ at last $3(i-1)$ partition jobs can be scheduled.*

Proof See Appendix. □

Lemma 10 *Jobs in each block E_i are processed one after another and between E_i and E_{i+1} exactly 3 partition jobs are scheduled for $i = 1, \dots, m-1$.*

Proof See Appendix. □

Based on the above considerations, we will prove the following theorem.

Theorem 2 *The problem $1|p_j(v) = a - b_j v|L_{\max}$ is strongly NP-hard.*

Proof Based on the given transformation from 3PP to DLEL and on Lemma 10 we construct a schedule π for DLEL, that is given as follows: $(E_1, X_1, E_2, X_2, E_3, \dots, E_i, X_i, E_{i+1}, \dots, X_{m-1}, E_m, X_m)$. Recall that blocks of the enforcer jobs are scheduled according to the EDD rule and the schedule of jobs within each E_i is immaterial and the sequence of jobs within each set X_i is arbitrary. To make the calculations easier, renumber the jobs in these sets, i.e., $X_i = \{3i-2, 3i-1, 3i\}$ for $i = 1, \dots, m$.

The further part of the proof is exactly the same as for Theorem 1. □

Note that any further relaxation of the problem $1|p_j(v) = a - b_j v|L_{\max}$ is polynomially solvable, namely $1|p_j(v) = a - bv|L_{\max}$ (the EDD rule, see Lemma 7) and $1|p_j(v) = a - b_j v, d_j = d|L_{\max}$ that is equivalent to $1|p_j(v) = a - b_j v|C_{\max}$ (see Lemma 8). Therefore, we also determine the boundary between polynomially solvable and NP-hard cases.

Since $1|p_j(v) = a - b_j v|L_{\max}$ is strongly NP-hard, thereby $1|p_j(v) = a - b_j v|\sum U_j$ is not less complex.

3.3 Problem equivalency

In the classical scheduling theory, the following problems $1||L_{\max}$ and $1|r_j|C_{\max}$ are equivalent with respect to the criterion value. Moreover, an algorithm solving problem $1||L_{\max}$ can be taken as an algorithm solving the problem $1|r_j|C_{\max}$. Now, we will show that this equivalency still holds in the presence of learning and aging, but if the corresponding job processing times are symmetric for the both phenomena.

Theorem 3 *The problems $1|p_j(v)|L_{\max}$ and $1|r'_j, p'_j(v)|C'_{\max}$ are equivalent in the following sense: the optimal schedules are inverse and the criterion values differ only by a constant, if $p_j(v)$ is a positive function of a job position and $p'_j(v) = p_j(n - v + 1)$ for $v, j = 1, \dots, n$.*

Proof First, a transformation from $1|p_j(v)|L_{\max}$ (LP) to $1|r'_j, p'_j(v)|C'_{\max}$ (CP) is given:

$$n' = n; \qquad r'_j = D - d_j; \quad j = 1, \dots, n,$$

where $D = \max_{j=1,\dots,n} d_j$. It is trivial to show that the transformation can be done in polynomial time.

Given a schedule π for the problem LP construct a schedule π' for the problem CP viewed from the reverse direction. It means that for a given permutation $\pi = \langle \pi(1), \pi(2), \dots, \pi(n) \rangle$, the corresponding permutation π' is defined as $\pi'(v) = \pi(n - v + 1)$ for $v = 1, \dots, n$. Since $p'_j(v) = p_j(n - v + 1)$ for $v, j = 1, \dots, n$, then the processing times of the jobs placed in the vth position in π' and in the $(n - v + 1)$th position in π are equal, i.e., $p'_{\pi'(v)}(v) = p_{\pi(n-v+1)}(n - v + 1)$ for $v = 1, \dots, n$.

Let S'_j denote the starting time of job j. First, we show that if for the given permutation π of the problem LP the following equality $L_{\pi(n-k+1)} = L_{\max}$ holds for job $\pi(n - k + 1)$, then in the corresponding permutation π' of the problem CP, job $\pi'(k)$ starts at its release date (i.e., $S'_{\pi'(k)} = r'_{\pi'(k)}$), $k = 1, \dots, n$.

Observe that for the given job $\pi(n - k + 1)$ the following equality $L_{\max} = L_{\pi(n-k+1)} \geq L_{\pi(n)} = C_{\pi(n)} - d_{\pi(n)} \geq C_{\pi(n)} - D$ holds, thus $C_{\pi(n-k+1)} - d_{\pi(n-k+1)} \geq C_{\pi(n)} - D$. Note also that $C_{\pi(n)} - C_{\pi(n-k+1)} = \sum_{i=n-k+2}^{n} p_{\pi(i)}(i)$ and on this basis:

$$D - d_{\pi(n-k+1)} \geq C_{\pi(n)} - C_{\pi(n-k+1)} = \sum_{i=n-k+2}^{n} p_{\pi(i)}(i),$$

$$r'_{\pi(n-k+1)} \geq \sum_{i=n-k+2}^{n} p'_{\pi(i)}(i),$$

$$r'_{\pi'(k)} \geq \sum_{i=1}^{k-1} p'_{\pi'(i)}(i) = C'_{\pi'(k-1)}.$$

Since $S'_{\pi'(k)} = \max\{r'_{\pi'(k)}, C'_{\pi'(k-1)}\}$, job $\pi'(k)$ starts at its release date.

On this basis, we prove that for the constructed permutations π and π' the optimal criterion values for the corresponding problems differ only by a constant. Thus, assume that $n - i + 1$ denotes a position of the first job in π for which the following equality $L_{\max} = L_{\pi(n-i+1)}$ holds. Therefore, we have $S'_{\pi'(i)} \geq r'_{\pi'(i)}$ for $i = k, \dots, n$. Thus, the criterion value calculated for CP is equal to

$$\begin{aligned} C'_{\max} &= r'_{\pi'(k)} + \sum_{i=k}^{n} p'_{\pi'(i)}(i) = D - d_{\pi(n-k+1)} + \sum_{i=k}^{n} p_{\pi(n-i+1)}(n - i + 1) \\ &= D + \sum_{i=1}^{n-k+1} p_{\pi(i)}(i) - d_{\pi(n-k+1)} = D + C_{\pi(n-k+1)} - d_{\pi(n-k+1)} = D + L_{\max}. \end{aligned}$$

Since the processing times of the jobs placed in appropriate positions in both schedules are equal, the criterion values calculated for both problems differ only by the constant D. Thus, we proved that both problems are equivalent. □

On this basis, we can easily prove the complexity of the following problems.

Corollary 1 *The problem* $1|r_j, p_j(v) = a_j(n+1-v)|C_{\max}$ *is strongly NP-hard.*

Proof The problem $1|p_j(v) = a_j v|L_{\max}$ is strongly NP-hard (Theorem 1), thus, on the basis of Theorem 3, the considered problem $1|r_j, p_j(v) = a_j(n+1-v)|C_{\max}$ is not less complex. □

Corollary 2 *The problem* $1|r_j, p_j(v) = a'_j + a_j v, a'_j = (c - a_j(n+1)), c > a_j n|C_{\max}$ *is strongly NP-hard.*

Proof On the basis of Theorem 2 and Theorem 3, in the similar manner as the previous proof. □

Note that we prove the strong NP-hardness of the problems with models that are even simpler than analyzed by Bachman and Janiak (2004).

4 Polynomially solvable cases

The problems defined in Sect. 2 are strongly NP-hard even if job processing times are described by simple linear functions, where the decreasing/increasing of a job processing time is different for each job. However, in this section, we prove that if the decrease/increase of a job processing time is the same for each job (i.e., $p_j(v) = a_j + f(v)$), then the considered problems can be solved optimally in polynomial time even if the function $f(v)$ is arbitrary. Therefore, we will determine the boundary between polynomially solvable and NP-hard cases.

The algorithms presented in this section solve the problems with the learning effect as well as with the aging effect.

Property 1 The problem $1|r_j, p_j(v) = a_j + f(v)|C_{\max}$ can be solved optimally in $O(n \log n)$ by scheduling jobs according to the non-decreasing order of their release dates (Earliest Release Dates—the ERD rule).

Property 2 The problem $1|p_j(v) = a_j + f(v)|L_{\max}$ can be solved optimally in $O(n \log n)$ steps by scheduling jobs according to the non-decreasing order of their due dates (Earliest Due Date—the EDD rule).

Since Property 1 and Property 2 can be proved by simple job interchanging technique the proofs are omitted.

It is well known that the problem $1||\sum U_j$ (with constant job processing times) can be solved optimally by Moore's Algorithm (Moore 1968). We will prove that this algorithm is still optimal for the problem $1|p_j(v) = a_j + f(v)|\sum U_j$.

Property 3 The problem $1|p_j(v) = a_j + f(v)|\sum U_j$ can be solved optimally in $O(n \log n)$ steps by Moore's Algorithm.

Proof The proof will be done using the inductive method in the similar manner as by Sturm (1970). Based on Property 2, we can note that there exists a schedule for $1|p_j(v) = a_j +$

Algorithm 1 Moore's Algorithm (MA)

```
1: Schedule the jobs in non-decreasing order of their due dates (EDD)
2:   If no jobs in the sequence are late go to Step 7
3:   Find the first late job, denote this job by $\alpha$
4:   Find a job $\beta$ such that $a_\beta = \max_{i=1,\dots,\alpha}\{a_i\}$
5:   Remove $\beta$ from the schedule and process it after all the jobs that
     are not late have been processed
6:   go to Step 2
7: The schedule is optimal
```

$f(v)|\sum U_j$ having no late jobs if and only if the schedule of jobs according to the non-decreasing order of their due dates (EDD) has no late jobs. On this basis, we will consider only EDD sequences. To simplify the proof, assume that such a sequence is $1, 2, \dots, n$ (if it is not a case we can renumber the jobs).

Assume that using Moore's Algorithm (Algorithm 1) we determine a subset $B = \{\beta_1, \dots, \beta_q\}$ of q late jobs. Suppose also that it is possible to choose from the set $J = \{1, \dots, n\}$ a subset $\Gamma = \{\gamma_1, \dots, \gamma_{q-1}\}$ of $q-1$ jobs, such that the remaining $n-q+1$ jobs $J\backslash\Gamma$ are not late. Thus, for all $i = 1, \dots, n$ the following inequality must hold:

$$d_i \geq \sum_{j=1}^{i}\big(a_j + f(j)\big) - \sum_{\gamma_j\in\Gamma_i} a_{\gamma_j} - \sum_{j=1}^{|\Gamma_i|} f(i-j+1), \tag{20}$$

where $\Gamma_i = \{\gamma_j : \gamma_j \leq i, \gamma_j \in \Gamma\}$ and $|\Gamma_i|$ is the cardinality of Γ_i. Without loss of generality we can also assume $\forall(i,j)\beta_i \neq \gamma_j$.

Using Moore's Algorithm (MA), we find the first late job α_1, i.e., that satisfies $d_{\alpha_1} < \sum_{j=1}^{\alpha_1} a_j + f(j)$ and $d_i \geq \sum_{j=1}^{i} a_j + f(j)$ for $i = 1, \dots, \alpha_1 - 1$. From the definition of Γ follows that there is at least one job $\gamma_j \in \Gamma_{\alpha_1}$, i.e., inequality (20) must hold. Let us choose an element δ_1 from Γ_{α_1} with $a_{\delta_1} = \max\{a_{\gamma_i} : \gamma_i \in \Gamma_{\alpha_1}\}$. On the other hand, MA chooses job β_1 ($a_{\beta_1} \geq a_{\alpha_1}$) that satisfies

$$\begin{aligned} d_{\alpha_1} \geq d_{\alpha_1 - 1} \geq \sum_{j=1}^{\alpha_1-1}\big(a_j + f(j)\big) &= \sum_{j=1}^{\alpha_1}\big(a_j + f(j)\big) - a_{\alpha_1} - f(\alpha_1) \\ &\geq \sum_{j=1}^{\alpha_1}\big(a_j + f(j)\big) - a_{\beta_1} - f(\alpha_1). \end{aligned}$$

Observe that if $\alpha_1 \neq \beta_1$, then job α_1 is no longer late, since job β_1 is skipped. It is easy to notice that $a_{\delta_1} \leq a_{\beta_1}$. Thus, there must be at least one job in Γ_{α_1} and $\delta_1 \in \Gamma_{\alpha_1}$.

Suppose now that there is at least l ($l < q$) jobs in Γ_{α_l} and we are able to choose among them l elements δ_i such that $a_{\delta_i} \leq a_{\beta_i}$ for $i = 1, \dots, l$.

Using MA we find job α_{l+1} (i.e., the first late job after l jobs are skipped) that satisfies

$$\begin{aligned} d_{\alpha_{l+1}} &< \sum_{j=1}^{\alpha_{l+1}}\big(a_j + f(j)\big) - \sum_{j=1}^{l} a_{\beta_j} - \sum_{j=1}^{l} f(\alpha_{l+1} - j + 1) \\ &\leq \sum_{j=1}^{\alpha_{l+1}}\big(a_j + f(j)\big) - \sum_{j=1}^{l} a_{\delta_j} - \sum_{j=1}^{l} f(\alpha_{l+1} - j + 1). \end{aligned}$$

From (20) follows that there must be at least $l+1$ jobs in $\Gamma_{\alpha_{l+1}}$ to satisfy

$$d_{\alpha_{l+1}} \geq \sum_{j=1}^{\alpha_{l+1}} \big(a_j + f(j)\big) - \sum_{\gamma_j \in \Gamma_{\alpha_{l+1}}} a_{\gamma_j} - \sum_{j=1}^{|\Gamma_{\alpha_{l+1}}|} f(\alpha_{l+1} - j + 1),$$

and $\delta_i \in \Gamma_{\alpha_{l+1}}$, $i = 1, \dots, l$. Thus, we find the $(l+1)$th element with $a_{\delta_{l+1}} = \max\{a_{\gamma_i} : \gamma_i \in \Gamma_{\alpha_{l+1}} \backslash \{\delta_j\}, j = 1, \dots, l\}$. On the other hand, MA finds β_{l+1} (i.e., the $(l+1)$th late job) with $a_{\beta_{l+1}} = \max_{\{} a_{\beta_i} : i = 1, \dots, \alpha_{l+1}, i \neq \beta_1, \dots, \beta_l\}$ and it is easy to notice that $a_{\delta_{l+1}} \leq a_{\beta_{l+1}}$. Therefore, there must be at least $l+1$ jobs in $\Gamma_{\alpha_{l+1}}$ and among them $l+1$ jobs δ_i with $a_{\delta_i} \leq a_{\beta_i}$ for $i = 1, \dots, l+1$. Concluding in the same way, we can show that when MA finds job α_q, then there must be at least q jobs in Γ_{α_q} and it contradicts the assumption $|\Gamma| = q - 1$. Thus MA finds the minimum number of late jobs for the considered scheduling problem. Note that the complexity of MA is $O(n \log n)$ if the algorithm is implemented with a special data structure. □

5 Conclusions

In this paper, we proved that the minimization of the maximum lateness or of the makespan with release dates is strongly NP-hard even if job processing times are described by simple linear functions dependent on a number of processed jobs (i.e., a job position in a sequence). Moreover, we showed that the minimization of the makespan with release dates is equivalent to the minimization of the maximum lateness if job processing times are described by functions dependent on the number of processed jobs and the functions are monotonically opposite for these problems.

The main conclusion concerning the proved strong NP-hardness of the considered problems is that the maximum acceptable simplification of job processing time functions (by their linearization) does not decrease the complexity of the considered problems. Therefore, it does not decrease the effort required to obtain optimal control decisions but it rather decreases their accuracy.

Finally, we also proved that the considered problems with an arbitrary functions describing decrease/increase of a job processing time are polynomially solvable if the functions are the same for each job. The proper algorithms were provided.

The future research will concern on the analysis of the scheduling problems with position dependent job processing times under additional constraints and different criteria (e.g., Leung et al. 2008; Shabtay and Steiner 2008; Steiner and Zhang 2011; Xu et al. 2010).

Acknowledgement This work was financially supported by the Polish Ministry of Science and Higher Education in years 2012–2013.

Appendix

Proof of Lemma 4 The proof will be done using the inductive method. At first observe that in the optimal schedule the blocks of the enforcer jobs are scheduled according to the EDD rule (Lemma 2) and the sequence of jobs within each block is immaterial (Lemma 1).

Let S_{E_i} denote the start time of the first job in the set E_i (for $i = 1, \dots, m$). Obviously if before jobs in E_1 any partition job is scheduled, then $S_{E_1} > d_{E_1}$. Now, we will show that if more than 3 jobs are scheduled before E_2, then all jobs from E_2 are late. Assume that at least 4 partition jobs are scheduled before E_2, then based on (11)–(13) and taking into consideration that $q \in X_2$ and $p_q(N+4) > M$, we have

$$\begin{aligned} C_{E_2} > S_{E_2} &> W_1 + V(X_1) + p_q(N+4) > W_1 + V_1 - (N+3)B - \frac{3}{4}B + p_q(N+4) \\ &= W_1 + V_1 + W_2 + p_q(N+4) - (N+3)B - \frac{3}{4}B - W_2 \\ &> W_1 + V_1 + W_2 + M - (N+3)B - \frac{3}{4}B - W_2 > W_1 + V_1 + W_2 = d_{E_2}. \end{aligned}$$

Since $S_{E_2} > d_{E_2}$, then all jobs in E_2 are late. Thus, we conclude that in the optimal solution of DAEL the number of the partition jobs before E_2 cannot be greater than 3.

Now we will show that if before E_3 more than 6 partition jobs are scheduled, then all jobs in E_3 are late. Assume we have at least 7 partition jobs scheduled before the enforcer jobs from E_3. From Lemma 3 follows that C_{E_3} is minimal if jobs are scheduled according to LPT rule. Thus, C_{E_3} is minimal if the partition jobs are scheduled first. However, from the previous considerations follows that the number of the partition jobs before E_2 cannot exceed 3 otherwise jobs from E_2 are always late. Therefore, jobs are scheduled as follows $(E_1, X_1, E_2, X_2, q, E_3, \dots)$, where $q \in X_3$, then we have

$$\begin{aligned} C_{E_3} > S_{E_3} &> \sum_{l=1}^{2}\big(W_l + V(X_l)\big) + p_q\big(2(N+3)+1\big) \\ &> \sum_{l=1}^{2}\left(W_l + V_l - l(N+3)B - \frac{3}{4}B\right) + W_3 + M - W_3 \\ &> \sum_{l=1}^{2}(W_l + V_l) + W_3 + M - \sum_{l=1}^{2}\left(l(N+3)B + \frac{3}{4}B\right) - W_3 \\ &> \sum_{l=1}^{2}(W_l + V_l) + W_3 = d_{E_3}. \end{aligned}$$

Since $S_{E_3} > d_{E_3}$, then all jobs from E_3 are late. Thus, in the optimal solution of DAEL, the number of the partition jobs before E_3 cannot be greater than 6.

In order to proceed inductively, consider the set E_i. Suppose that before each set E_l $3(l-1)$ partition jobs are scheduled (for $l = 1, \dots, i-1$) and before E_i more than $3(i-1)$ jobs are scheduled ($i = 2, \dots, m$). On this basis and taking into consideration (11)–(13), we have

$$\begin{aligned} C_{E_i} > S_{E_i} &> \sum_{l=1}^{i-1}\big(W_l + V(X_l)\big) + p_q\big((i-1)(N+3)+1\big) \\ &> \sum_{l=1}^{i-1}(W_l + V_l) + W_i + M - \sum_{l=1}^{i-1}\left(l(N+3)B + \frac{3}{4}B\right) - W_{E_i} \end{aligned}$$

$$> \sum_{l=1}^{i-1}(W_l + V_l) + W_i = d_{E_i},$$

for $i = 2, \dots, m$ and $q \in X_{i+1}$. Thus, we conclude that the number of the partition jobs before E_i (for $i = 1, \dots, m$) cannot be greater than $3(i-1)$ otherwise all jobs in E_i are late. □

Proof of Lemma 5 From Lemma 3 follows that the completion time of a job is minimal if before it jobs are scheduled according to LPT rule. On the other hand from Lemma 4 follows that in the optimal solution the number of the partition jobs before E_i cannot be greater than $3(i-1)$ for $i = 1, \dots, m$, otherwise all jobs from E_i are late. Under this assumption the completion time C_{X_m} of the last job in the set X_m is minimal if the schedule of jobs π^* is as follows: $(E_1, X_1, E_2, X_2, E_3, \dots, E_i, X_i, E_{i+1}, \dots, X_{m-1}, E_m, X_m)$. Thus, the completion time of the last job in X_m for π^* is equal to:

$$C_{X_m}\big(\pi^*\big) = \sum_{l=1}^{m}\big(W_l + V(X_l)\big).$$

Now we will show that the optimal schedule is in the form of π^*. To prove it we will use the inductive principle.

Obviously before E_1 no partition jobs are scheduled. First assume that before an enforcer job from E_2 at last 2 partition jobs are scheduled (i.e., $E_1, 1, 2, e_{N+1}, 3, e_{N+2}, \dots, e_{2N}, \dots$). Note that jobs from E_i are indistinguishable ($i = 1, \dots, m$). Observe that in this case the value $V(X_1)$ (where $X_1 = \{1, 2, 3\}$) increases by a_3 and W_2 (where $E_2 = \{e_{N+1}, e_{N+2}, \dots, e_{2N}\}$) decreases by a_E. On this basis and taking into consideration (11) and (13), the completion time of the last job in E_2 can be estimated as follows:

$$\begin{aligned}
C_{E_2} &> W_1 + V(X_1) + a_3 + W_2 - a_E \\
&> W_1 + V_1 + W_2 + M - (N+3)B - \frac{3}{4}B - N \\
&> W_1 + V_1 + W_2 = d_{E_2}.
\end{aligned}$$

Since $C_{E_2} > d_{E_2}$, then the last scheduled job in E_2 is late. Observe that this value is greater if more enforcer jobs from E_2 are moved before job 3. Thus, E_2 is not late if jobs from E_2 are scheduled one after another, i.e., $(E_1, 1, 2, E_2, 3, \dots)$. However, if before E_2 at last 2 partition jobs are scheduled, then C_{X_m} can be estimated as follows:

$$\begin{aligned}
C_{X_m} &> C_{X_m}\big(\pi^*\big) + a_3 - N a_E \\
&> \sum_{l=1}^{m}(W_l + V_l) + M - \sum_{l=1}^{m}\bigg(l(N+3)B + \frac{3}{4}B\bigg) - N \\
&> \sum_{l=1}^{m}(W_l + V_l) = D.
\end{aligned}$$

Since $C_{X_m} > D$, then $L_{\max} > 0$. Taking it into consideration and based on Lemma 4, we conclude that in the optimal solution jobs from E_2 are processed one after another and 3 partition jobs are scheduled between E_1 and E_2, otherwise $L_{\max} > 0$.

Now we will show that if before E_3 less than 6 partition jobs are scheduled, then $L_{\max} > 0$. Assume we have at last 5 partition jobs scheduled before an enforcer job from E_3. First we will show that in the optimal solution of DAEL jobs from E_3 have to be processed one after another. From the previous considerations follows that the schedule is given as follows: $(E_1, X_1, E_2, 4, 5, e_{2N+1}, 6, e_{2N+2}, \ldots, e_{3N}, \ldots)$. Thus, the completion time of the last job in E_3 is equal to:

$$\begin{aligned} C_{E_3} &= \sum_{l=1}^{2}\bigl(W_l + V(X_l)\bigr) + a_6 + W_2 - a_E \\ &> \sum_{l=1}^{2}(W_l + V_l) + W_3 + M - \sum_{l=1}^{2}\biggl(l(N+3)B + \frac{3}{4}B\biggr) - N \\ &> \sum_{l=1}^{2}(W_l + V_l) + W_3 = d_{E_3}. \end{aligned}$$

Since $C_{E_3} > d_{E_3}$, then the last scheduled job in E_3 is late. Observe that this value is greater if more enforcer jobs from E_3 are moved before job 6. Thus, E_3 is not late if jobs from E_3 are scheduled one after another, i.e., $(E_1, X_1, E_2, 4, 5, E_3, 6, \ldots)$. However, if before E_3 at last 5 jobs are scheduled, then C_{X_m} can be estimated as follows:

$$\begin{aligned} C_{X_m} &> C_{X_m}\bigl(\pi^*\bigr) + a_6 - Na_E \\ &> \sum_{l=1}^{m}(V_l + W_l) + M - \sum_{l=1}^{m}\biggl(l(N+3)B + \frac{3}{4}B\biggr) - N \\ &> \sum_{l=1}^{m}(V_l + W_l) = D. \end{aligned}$$

Taking it into consideration and based on Lemma 4, we conclude that in the optimal solution jobs from E_3 are processed one after another and 3 partition jobs are scheduled between E_2 and E_3, otherwise $L_{\max} > 0$.

In order to proceed inductively consider the set E_i. Suppose that between E_l and E_{l+1} exactly 3 partition jobs are scheduled ($l = 1, \ldots, i-2$). Assume that before E_i at last $3(i-1) - 1$ partition jobs are scheduled. From the previous considerations follows that the schedule is given as follows: $(E_1, X_1, E_2, X_2, E_3, \ldots, E_{i-1}, 3(i-1) - 2, 3(i-1) - 1, e_{iN+1}, 3(i-1), e_{iN+2}, \ldots, e_{(i+1)N}, \ldots)$. On this basis and taking into consideration (11) and (13), the completion time of the last job in E_i can be estimated as follows:

$$\begin{aligned} C_{E_i} &> \sum_{l=1}^{i-1}\bigl(W_l + V(X_l)\bigr) + a_{3(i-1)} + W_i - Na_E \\ &> \sum_{l=1}^{i-1}(W_l + V_l) + W_i + M - \sum_{l=1}^{i-1}\biggl(l(N+3)B + \frac{3}{4}B\biggr) - N \\ &> \sum_{l=1}^{i-1}(W_l + V_l) + W_i = d_{E_i}, \end{aligned}$$

for $i = 2, \ldots, m$. Thus, jobs from E_i must be scheduled one after another, i.e., $(E_1, X_1, \ldots, E_{i-1}, 3(i-1)-2, 3(i-1)-1, E_i, 3(i-1), \ldots)$, otherwise the last job in E_i is late (for $i = 1, \ldots, m$). However, if before E_i at last $3(i-1)$ jobs are scheduled, then C_{X_m} can be estimated as follows:

$$\begin{aligned} C_{X_m} &> C_{X_m}\left(\pi^*\right) + a_{3(i-1)} - Na_E \\ &> \sum_{l=1}^{m}(V_l + W_l) + M - \sum_{l=1}^{m}\left(l(N+3)B + \frac{3}{4}B\right) - N \\ &> \sum_{l=1}^{m}(V_l + W_l) = D. \end{aligned}$$

for $i = 1, \ldots, m-1$.

Thus, we conclude that in the optimal solution jobs from the set E_i are scheduled one after another and between E_i and E_{i+1} exactly 3 partition jobs are scheduled (for $i = 1, \ldots, m-1$), otherwise $L_{\max} > 0$. □

Proof of Lemma 9 The proof will be done using the inductive method in the similar manner as the proof of Lemma 4.

At first observe that in the optimal schedule the blocks of the enforcer jobs are scheduled according to the EDD rule (Lemma 7) and the sequence of jobs within each block is immaterial (Lemma 6). Note also that C_{E_i} is minimal if before E_i partition jobs are scheduled first, since $b_j < b_E$ for $j = 1, \ldots, 3m$ and $i = 1, \ldots, m$ (Lemma 8). However, the sum of processing times of the enforcer jobs E_i for an arbitrary schedule π (denoted as W_i') is always greater than the sum of processing times of the enforcer jobs E_i for schedule π^* (denoted as W_i) decreased by $b_E mN(N+3)$, i.e., $W_i' > W_i - b_E mN(N+3)$ for $i = 1, \ldots, m$. Note that $m(N+3)$ is the maximum possible position in a schedule for DLEL.

Assume that before E_1 at least one partition job is scheduled. Thus, based on (19), the completion time of the last job in E_1 can be estimated as follows:

$$C_{E_1} > p_q(1) + W_1 - mN(N+3)b_E > W_1 + a - M - mN(N+3)b_E > W_1 = d_{E_1}.$$

Therefore, we conclude that in the optimal solution of DLEL any partition job can be scheduled before E_1, otherwise $L_{\max} > 0$.

Now, we will show that if more than 3 jobs are scheduled before E_2, then the last job from E_2 is late. Assume that at least 4 partition jobs are scheduled before E_2. Based on (17) and (19), we have

$$\begin{aligned} C_{E_2} &> W_1 + V(X_1) + p_q(N+4) + W_2 - mN(N+3)b_E \\ &> W_1 + V_1 - (N+3)B - \frac{3}{4}B + W_2 + a - M(N+4) - mN(N+3)b_E \\ &> W_1 + V_1 + W_2 + a - M(N+4) - mN(N+3)b_E - (N+3)B - \frac{3}{4}B \\ &> W_1 + V_1 + W_2 = d_{E_2}, \end{aligned}$$

where $q \in X_2$. Since $C_{E_2} > d_{E_2}$, then the last job in E_2 is late. Thus, we conclude that in the optimal solution of DLEL the number of the partition jobs before E_2 cannot be greater than 3.

Now we will show that if before E_3 more than 6 partition jobs are scheduled, then all jobs in E_3 are late. Assume we have at least 7 partition jobs scheduled before the enforcer jobs from E_3. From Lemma 8 follows that C_{E_3} is minimal if the partition jobs are scheduled first. However, from the previous considerations follows that the number of the partition jobs before E_2 cannot exceed 3 otherwise $L_{\max} > 0$. Therefore, jobs are scheduled as follows $(E_1, X_1, E_2, X_2, q, E_3, \dots)$ (where $q \in X_3 = \{7, 8, 9\}$) and we have

$$\begin{aligned} C_{E_3} &> \sum_{l=1}^{2}\big(W_l + V(X_l)\big) + p_q\big(2(N+3)+1\big) + W_3 - mN(N+3)b_E \\ &> \sum_{l=1}^{2}(W_l + V_l) + W_3 + a - M\big(2(N+3)+1\big) - mN(N+3)b_E \\ &\quad - \sum_{l=1}^{2}\left(l(N+3)B + \frac{3}{4}B\right) > \sum_{l=1}^{2}(W_l + V_l) + W_3 = d_{E_3}, \end{aligned}$$

Thus, in the optimal solution of DAEL, the number of the partition jobs before E_3 cannot be greater than 6.

In order to proceed inductively, consider the set E_i. Suppose that before each set E_l $3(l-1)$ partition jobs are scheduled (for $l = 1, \dots, i-1$) and before E_i more than $3(i-1)$ jobs are scheduled ($i = 2, \dots, m$). On this basis and taking into consideration (17) and (19), we have

$$\begin{aligned} C_{E_i} &> \sum_{l=1}^{i-1}\big(W_l + V(X_l)\big) + p_q\big((i-1)(N+3)+1\big) + W_i - mN(N+3)b_E \\ &> \sum_{l=1}^{i-1}(W_l + V_l) + W_i + a - M\big((i-1)(N+3)+1\big) - mN(N+3)b_E \\ &\quad - \sum_{l=1}^{i-1}\left(l(N+3)B + \frac{3}{4}B\right) > \sum_{l=1}^{i-1}(W_l + V_l) + W_i = d_{E_i}, \end{aligned}$$

for $i = 2, \dots, m$ and $q \in X_i$. Thus, we conclude that the number of the partition jobs before E_i (for $i = 1, \dots, m$) cannot be greater than $3(i-1)$ otherwise all jobs in E_i are late. □

Proof of Lemma 10 The proof will be done in the similar manner as the proof of Lemma 5.

From Lemma 8 follows the completion time of a job is minimal if jobs before it are scheduled according to the non-decreasing order of b_j. On the other hand from Lemma 9 follows that in the optimal solution the number of the partition jobs before E_i cannot be greater than $3(i-1)$ for $i = 1, \dots, m$, otherwise at last one job in E_i is late. Under this assumption the completion time C_{X_m} of the last job in the set X_m is minimal if the schedule of jobs π^* is as follows: $(E_1, X_1, E_2, X_2, E_3, \dots, E_i, X_i, E_{i+1}, \dots, X_{m-1}, E_m, X_m)$. Thus, the completion time of the last job in X_m for π^* is equal to:

$$C_{X_m}(\pi^*) = \sum_{i=1}^{m}\big(W_l + V(X_l)\big).$$

Now we will show that the optimal schedule is in the form of π^*. To prove it we will use the inductive principle.

From Lemma 9 follows that before E_1 no partition jobs are scheduled. First assume that before an enforcer job from E_2 at last 2 partition jobs are scheduled (i.e., $E_1, 1, 2, e_{N+1}, 3, e_{N+2}, \ldots, e_{2N}, \ldots$). Note that jobs from E_i are indistinguishable ($i = 1, \ldots, m$). Observe that in this case the value $V(X_1)$ decreases by b_3 ($q \in X_1 = \{1, 2, 3\}$) and W_2 increases by b_E, where $E_2 = \{e_{N+1}, e_{N+2}, \ldots, e_{2N}\}$. On this basis and taking into consideration (17)–(19), the completion time of the last job in E_2 can be estimated as follows:

$$\begin{aligned} C_{E_2} &= W_1 + V(X_1) - b_3 + W_2 + b_E \\ &> W_1 + V_1 + W_2 + MN - (N+3)B - \frac{3}{4}B \\ &> W_1 + V_1 + W_2 = d_{E_2}. \end{aligned}$$

Since $C_{E_2} > d_{E_2}$, then the last scheduled job in E_2 is late. Observe that C_{E_2} is greater if more enforcer jobs from E_2 are moved before job 3. Thus knowing that $V(X_i) < V_i + i(N+3)B$, any job from E_2 is not late if jobs from E_2 are scheduled one after another, i.e., $(E_1, 1, 2, E_2, 3, \ldots)$:

$$\begin{aligned} C_{E_2} &= W_1 + V(X_1) - \big(a - (N+3)b_3\big) + W_2 + Nb_E \\ &< W_1 + V_1 + (N+3)B - a + (N+3)b_3 + W_2 + Nb_E \\ &= d_E - a + (N+3)b_E < d_{E_2}. \end{aligned}$$

However, if before E_2 at last 2 partition jobs are scheduled, then C_{X_m} can be estimated as follows:

$$\begin{aligned} C_{X_m} &= C_{X_m}\big(\pi^*\big) - Nb_3 + Nb_E \\ &> \sum_{l=1}^{m}(W_l + V_l) + MN^2 - \sum_{l=1}^{m}\bigg(l(N+3)B + \frac{3}{4}B\bigg) \\ &> \sum_{l=1}^{m}(W_l + V_l) = D. \end{aligned}$$

Since $C_{X_m} > D$, then $L_{\max} > 0$. Taking it into consideration and based on Lemma 9, we conclude that in the optimal solution jobs from E_2 are processed one after another and 3 partition jobs are scheduled between E_1 and E_2, otherwise $L_{\max} > 0$.

Now we will show that if before E_3 less than 6 partition jobs are scheduled, then $L_{\max} > 0$. Assume we have at last 5 partition jobs scheduled before an enforcer job from E_3. First we will show that in the optimal solution of DLEL jobs from E_3 have to be processed one after another. From the previous considerations follows that the schedule is given as follows: $(E_1, X_1, E_2, 4, 5, e_{2N+1}, 6, e_{2N+2}, \ldots, e_{3N}, \ldots)$. Thus, the completion time of the last job in E_3 can be estimated as follows:

$$\begin{aligned} C_{E_3} &= \sum_{l=1}^{2}\big(W_l + V(X_l)\big) - b_6 + W_3 + b_E \\ &> \sum_{l=1}^{2}(W_l + V_l) + W_3 + MN - \sum_{l=1}^{2}\bigg(l(N+3)B + \frac{3}{4}B\bigg) > d_{E_3}. \end{aligned}$$

Since $C_{E_3} > d_{E_3}$, then the last scheduled job in E_3 is late. Observe that this value is greater if more enforcer jobs from E_3 are moved before job 6. Thus, E_3 is not late if jobs from E_3 are scheduled one after another, i.e., $(E_1, X_1, E_2, 4, 5, E_3, 6, \ldots)$. However, if before E_3 at last 5 jobs are scheduled, then C_{X_m} can be estimated as follows:

$$\begin{aligned} C_{X_m} &= C_{X_m}\left(\pi^*\right) - Nb_6 + Nb_E \\ &> \sum_{l=1}^{m}(V_l + W_l) + MN^2 - \sum_{l=1}^{m}\left(l(N+3)B + \frac{3}{4}B\right) \\ &> \sum_{l=1}^{m}(V_l + W_l) = D. \end{aligned}$$

Taking it into consideration and based on Lemma 9, we conclude that in the optimal solution jobs from E_3 are processed one after another and 3 partition jobs are scheduled between E_2 and E_3, otherwise $L_{\max} > 0$.

In order to proceed inductively consider the set E_i. Suppose that between E_l and E_{l+1} exactly 3 partition jobs are scheduled ($l = 1, \ldots, i-2$). Assume that before E_i at last $3(i-1)-1$ partition jobs are scheduled. From the previous considerations follows that the schedule is given as follows: $(E_1, X_1, E_2, X_2, E_3, \ldots, E_{i-1}, 3(i-1)-2, 3(i-1)-1, e_{iN+1}, 3(i-1), e_{iN+2}, \ldots, e_{(i+1)N}, \ldots)$. On this basis and taking into consideration (17)–(19), the completion time of the last job in E_i can be estimated as follows:

$$\begin{aligned} C_{E_i} &= \sum_{l=1}^{i-1}\left(W_l + V(X_l)\right) - b_{3(i-1)} + W_i + b_E \\ &> \sum_{l=1}^{i-1}(W_l + V_l) + W_i + MN - \sum_{l=1}^{i-1}\left(l(N+3)B + \frac{3}{4}B\right) > d_{E_i}, \end{aligned}$$

for $i = 2, \ldots, m$. Thus, jobs from E_i must be scheduled one after another, i.e., $(E_1, X_1, \ldots, E_{i-1}, 3(i-1)-2, 3(i-1)-1, E_i, 3(i-1), \ldots)$, otherwise the last job in E_i is late (for $i = 1, \ldots, m$). However, if before E_i at last $3(i-1)$ jobs are scheduled, then C_{X_m} can be estimated as follows:

$$\begin{aligned} C_{X_m} &= C_{X_m}\left(\pi^*\right) - Nb_{3(i-1)} + Nb_E \\ &> \sum_{l=1}^{m}(V_l + W_l) + MN^2 - \sum_{l=1}^{m}\left(l(N+3)B + \frac{3}{4}B\right) > D, \end{aligned}$$

for $i = 1, \ldots, m-1$.

Thus, we conclude that in the optimal solution jobs from the set E_i are scheduled one after another and between E_i and E_{i+1} exactly 3 partition jobs are scheduled (for $i = 1, \ldots, m-1$), otherwise $L_{\max} > 0$. □

References

Adler, P. S., & Clark, K. B. (1991). Behind the learning curve: a sketch of the learning process. *Management Science*, *37*, 267–281.

Arimoto, S., Kawamura, S., & Miyazaki, F. (1984). Bettering operations of robots by learning. *Journal of Robotic Systems*, *1*, 123–140.

Bachman, A., & Janiak, A. (2004). Scheduling jobs with position dependent processing times. *The Journal of the Operational Research Society*, *55*, 257–264.

Biskup, D. (1999). Single-machine scheduling with learning considerations. *European Journal of Operational Research*, *115*, 173–178.

Biskup, D. (2008). A state-of-the-art review on scheduling with learning effects. *European Journal of Operational Research*, *188*, 315–329.

Buşoniu, L., Babuška, R., & De Schutter, B. (2008). A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man and Cybernetics. Part C, Applications and Reviews*, *38*, 156–172.

Carlson, J. G., & Rowe, R. G. (1976). How much does forgetting cost? *Industrial Engineering*, *8*, 40–47.

Cheng, T. C. E., Ding, Q., & Lin, B. M. T. (2004). A concise survey of scheduling with time-dependent processing times. *European Journal of Operational Research*, *152*, 1–13.

Cheng, T. C. E., Lee, W.-C., & Wu, C.-C. (2010). Single-machine scheduling with deteriorating functions for job processing times. *Applied Mathematical Modelling*, *34*, 4171–4178.

Cheng, T. C. E., & Wang, G. (2000). Single machine scheduling with learning effect considerations. *Annals of Operations Research*, *98*, 273–290.

Cheng, T. C. E., Wu, C.-C., & Lee, W.-C. (2008). Some scheduling problems with deteriorating jobs and learning effects. *Computers & Industrial Engineering*, *54*, 972–982.

Cochran, E. B. (1960). New concepts of the learning curve. *The Journal of Industrial Engineering*, *11*, 317–327.

Dababneh, A. J., Swanson, N., & Shell, R. L. (2001). Impact of added rest breaks on the productivity and well being of workers. *Ergonomics*, *44*, 164–174.

Eilon, S. (1964). On a mechanistic approach to fatigue and rest periods. *International Journal of Production Research*, *3*, 327–332.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: a guide to the theory of NP-completeness*. San Francisco: Freeman.

Gawiejnowicz, S. (1996). A note on scheduling on a single processor with speed dependent on a number of executed jobs. *Information Processing Letters*, *57*, 297–300.

Gawiejnowicz, S. (2008). *Time-dependent scheduling*. Berlin: Springer.

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

Holzer, H. P., & Riahi-Belkaoui, A. (1986). *The learning curve: a management accounting tool*. Westport: Quorum Books.

Jaber, Y. M., & Bonney, M. (1999). The economic manufacture/order quantity (EMQ/EOQ) and the learning curve: Past, present, and future. *International Journal of Production Economics*, *59*, 93–102.

Janiak, A., & Rudek, R. (2010). Scheduling jobs under an aging effect. *The Journal of the Operational Research Society*, *61*, 1041–1048.

Janiak, A., & Rudek, R. (2011). A note on the learning effect in multi-agent optimization. *Expert Systems With Applications*, *38*, 5974–5980.

Keachie, E. C., & Fontana, R. J. (1966). Production lot sizing under a learning effect. *Management Science*, *13*, 102–108.

Kerzner, H. (1998). *Project management: a system approach to planning, scheduling, and controlling*. New York: Wiley.

Kuo, W.-H., & Yang, D.-L. (2008). Minimizing the makespan in a single-machine scheduling problem with the cyclic process of an aging effect. *The Journal of the Operational Research Society*, *59*, 416–420.

Lee, W.-C., & Lai, P.-J. (2011). Scheduling problems with general effects of deterioration and learning. *Information Sciences*, *181*, 1164–1170.

Lee, W.-C., Wu, C.-C., & Hsu, P.-H. (2010). A single-machine learning effect scheduling problem with release times. *Omega*, *38*, 3–11.

Leung, J. Y.-T., Li, H., & Pinedo, M. (2008). Scheduling orders on either dedicated or flexible machines in parallel to minimize total weighted completion time. *Annals of Operations Research*, *159*, 107–123.

Li, C.-L., & Cheng, T. C. E. (1994). An economic production quantity model with learning and forgetting considerations. *Production and Operations Management*, *3*, 118–132.

Lien, T. K., & Rasch, F. O. (2001). Hybrid automatic-manual assembly systems. *Annals of the CIRP*, *50*, 21–24.

Mandich, N. V. (2003). Overview of surface preparation of metals prior to finishing: Part 2. *Metal Finishing*, *101*, 33–58.

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

Mosheiov, G. (2001). Parallel machine scheduling with a learning effect. *The Journal of the Operational Research Society*, *52*, 1–5.

Roberts, P. (1983). A theory of the learning process. *The Journal of the Operational Research Society*, *34*, 71–79.

Rudek, A., & Rudek, R. (2011). A note on optimization in deteriorating systems using scheduling problems with the aging effect and resource allocation models. *Computers & Mathematics With Applications*, *62*, 1870–1878.

Shabtay, D., & Steiner, G. (2008). The single-machine earliness-tardiness scheduling problem with due date assignment and resource-dependent processing times. *Annals of Operations Research*, *159*, 25–40.

Steiner, G., & Zhang, R. (2011). Minimizing the weighted number of tardy jobs with due date assignment and capacity-constrained deliveries. *Annals of Operations Research*, *191*, 171–181.

Stanford, M., & Lister, P. M. (2004). Investigation into the relationship between tool-wear and cutting environments when turning EN32 steel. *Industrial Lubrication and Tribology*, *56*, 114–121.

Sturm, L. B. J. M. (1970). A simple optimality proof of Moore's sequencing algorithm. *Management Science*, *17*, 116–118.

Wang, J.-B., & Wang, M.-Z. (2011). Worst-case behavior of simple sequencing rules in flow shop scheduling with general position-dependent learning effects. *Annals of Operations Research*. *191*, 155–169.

Webb, G. K. (1994). Integrated circuit (IC) pricing. *High Technology Management Research*, *5*, 247–260.

Whiteson, S., & Stone, P. (2004). Adaptive job routing and scheduling. *Engineering Applications of Artificial Intelligence*, *17*, 855–869.

Wright, T. P. (1936). Factors affecting the cost of airplanes. *Journal of the Aeronautical Sciences*, *3*, 122–128.

Wu, C.-C., Lee, W.-C., & Chen, T. (2007). Heuristic algorithms for solving the maximum lateness scheduling problem with learning considerations. *Computers & Industrial Engineering*, *52*, 124–132.

Xu, K., Feng, Z., & Ke, L. (2010). A branch and bound algorithm for scheduling jobs with controllable processing times on a single machine to meet due dates. *Annals of Operations Research*, *181*, 303–324.

Yang, D.-L., & Kuo, W.-H. (2009). Single-machine scheduling with both deterioration and learning effects. *Annals of Operations Research*, *172*, 315–327.

Yang, S.-J., & Yang, D.-L. (2010). Minimizing the makespan on single-machine scheduling with aging effect and variable maintenance activities. *Omega*, *38*, 528–533.

Yang, S.-J., Yang, D.-L., & Cheng, T. C. E. (2010). Single-machine due-window assignment and scheduling with job-dependent aging effects and deteriorating maintenance. *Computers & Operations Research*, *37*, 1510–1514.

Yelle, L. E. (1979). The learning curve: historical review and comprehensive study. *Decision Sciences*, *10*, 302–328.

Zhang, X., Yan, G., Huang, W., & Tang, G. (2011). Single-machine scheduling problems with time and position dependent processing times. *Annals of Operations Research*, *186*, 345–356.