Int. J. Production Economics 117 (2009) 142–149

Contents lists available at ScienceDirect

Int. J. Production Economics

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

A note on due-date assignment and single machine scheduling with a learning/aging effect

Pei-Chann Chang [a,*], Shih-Hsin Chen [b], V. Mani [c]

[a] Department of Information Management, Yuan Ze University, 135, Yuan-Tung Road, Chung-Li, Tao-Yuan 32026, Taiwan, ROC
[b] Department of Electronic Commerce Management, Nanhua University, 32, Chungkeng, Dalin Chiayi 62248, Taiwan, ROC
[c] Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India

ARTICLE INFO

Article history:
Received 12 March 2007
Accepted 10 October 2008
Available online 22 October 2008

Keywords:
Scheduling
Single-machine
Learning effect
Due-date assignment

ABSTRACT

This paper considers the learning/aging effect in an n job single machine scheduling problem with common due date. The objective is to determine the optimal common due date and the optimal sequence of jobs that minimizes a cost function in the presence of learning/aging effect. The cost function depends on the individual job earliness and tardiness values; i.e., $\sum_{j=1}^{n}\{E_{[j]}+T_{[j]}\}$. This is a well-known problem when the learning/aging effect is not considered and it is shown in earlier studies that there are more than one optimal sequence and optimal common due dates. It is shown in earlier studies that there are 2^{r-1} optimal sequences to this problem if n is odd, and 2^r optimal sequences if n is even. The value of r is $(n+1)/2$ if n is odd, and the value of r is $n/2$ if n is even. In this paper, we derive two bounds B_α and B^*_α for the learning index α. We show that when $B_\alpha<\alpha<0$, then the optimal sequence is unique and provide an $O(n\log n)$ algorithm to obtain this unique optimal sequence and the optimal common due date. We also show that when $\alpha<B^*_\alpha$, the optimal sequence is obtained by arranging the longest job in first position and the rest of the jobs in SPT order. Similarly, we derive two bounds A_α and A^*_α for the aging index α. We show that when $0<\alpha<A_\alpha$, then the optimal sequence is unique and provide an $O(n\log n)$ algorithm to obtain this unique optimal sequence and the optimal common due date. We also show that when $\alpha>A^*_\alpha$, the optimal sequence is obtained by arranging the jobs in LPT order. We also present a numerical example for ease of understanding.

1. Introduction

In this paper, we consider the problem of scheduling a set of n independent jobs on a single machine with a learning/aging effect. All the jobs are assigned a common due date. The objective is to find the optimal common due date and the optimal sequence of jobs that jointly minimizes a total cost function. The cost function depends on the individual job earliness and tardiness values. This is a well-known problem (when the learning/aging effect is not considered) and is studied by many researchers. Some important studies in this area are: Cheng and Gupta (1989), Cheng (1984, 1987, 1991), Kanet (1981), and Panwalkar et al. (1982). Cheng (1992) has shown that the optimal sequence has a V-shape property with respect to processing times; i.e., the jobs are arranged in non-increasing order up to the position r in the sequence, and are arranged in non-decreasing order after the position r. The value of r is $(n+1)/2$ if n is odd, and the value of r is $n/2$ if n is even. In this study (Cheng, 1992), it is shown that the optimal sequence is not unique and we get more than one sequence for which the cost function has the same value. In fact it is shown in Cheng (1992) that (when the learning effect is not considered) there are 2^{r-1} optimal sequences to this problem if n is odd, and 2^r

* Corresponding author. Tel.: +886 3 4638800x2305; fax: +886 3 4638907.
E-mail address: iepchang@saturn.yzu.edu.tw (P.-C. Chang).

0925-5273/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.ijpe.2008.10.004

optimal sequences if n is even. In the single machine scheduling problems discussed in Cheng and Gupta (1989), Cheng (1984, 1987, 1991, 1992), Kanet (1981) and Panwalkar et al. (1982), the processing time of a job is assumed to be a constant. In other words, the processing time of a job is independent of its position in the sequence.

Learning effect: The learning effect can arise in scheduling of jobs due to the fact that workers are processing the same type of jobs on the same machine. Hence, it is possible for workers to improve their performance and so the processing time of a job will reduce because of the learning. Biskup (1999) is the first to study the effect of learning for the case of single machine scheduling problem. Mosheiov (2001a, b) considered the learning effect on a single and parallel identical machines with the objective of minimizing the flow-time. Lee and Wu (2004) considered the learning effect in a two machine flowshop scheduling with the objective to find the sequence of jobs that minimize the total completion time. A branch and bound technique is presented in Lee and Wu (2004) and to improve the efficiency of the branch and bound technique, a heuristic algorithm is also presented in Lee and Wu (2004). It is shown by Wang and Xia (2005) that the classical Johnson's rule is not the optimal solution to minimize the makespan for the problem of flowshop scheduling with a learning effect. Various issues related to scheduling problems with a learning effect are also studied by Mosheiov and Sidney (2005), Biskup and Simons (2004), Bachman and Janiak (2004), and Kuo and Yang (2006). The problem of minimizing the total tardiness on one machine is shown to be NP-hard by Du and Leung (1990). Another single machine scheduling problem studied by Cheng et al. (2003) in which the processing times of jobs decrease in a piecewise linear fashion is a good approximation to study the learning effect. Cheng et al. (1996) also studied this problem in the case of compressible processing times. Recent studies (Wang, 2006, 2007; Wang and Cheng, 2007) present the effect of learning along with deterioration in single machine scheduling problem. The time dependent learning effect in single machine scheduling is discussed by Wang et al. (2008). A survey of scheduling with time dependent processing times presented by Cheng et al. (2004) provide a framework to show how the time dependent processing time problems have been generalized from the classical scheduling theory.

Aging effect: Aging effect is the opposite of learning. In learning the actual processing time of a job decreases as a function of its position in the sequence. In aging effect the actual processing time of a job increases as a function of its position in the sequence. This aging effect is due to the fact that the production facility becomes less efficient (Mosheiov, 2001b).

2. Problem formulation

In this paper, we consider the single machine scheduling problem presented by Cheng (1992) with a learning effect. A set of n independent jobs to be processed on a continuously available single machine. The machine can process only one job at a time and job splitting and inserting idle times are not permitted. Each job has a normal processing time p_j ($j = 1, 2, \ldots, n$) if they are at the first position in the sequence. The sequence is the order in which the jobs are processed in the machine. The jobs are numbered according to shortest normal processing time rule, i.e., $p_1 \leqslant p_2 \leqslant \cdots \leqslant p_n$. Because of the learning effect, the processing time of a job depends on its position in the sequence. Hence, the processing time of a job is given as

$$p_{jl} = p_j l^{\alpha} \tag{1}$$

In the above equation, p_{jl} is the processing time of job j if it is in position l of the sequence, and α is the learning index and $\alpha < 0$. From Eq. (1), we see that $p_{j1} > p_{j2} > p_{j3} > \cdots > p_{jn}$. For example, if $p_j = 3$ and $\alpha = -0.515$, then $p_{j1} = 3$, $p_{j2} = 2.0994$, $p_{j3} = 1.7037$, $p_{j4} = 1.4691$, $p_{j5} = 1.3095$, and so on. We can see that the actual processing time of job j decreases when its position in the sequence increases. All the jobs are available at the same time and are assigned a common due-date $d_j = k$, for $j = 1, 2, \ldots, n$, and $k \geqslant 0$ is the common due-date value. It can be easily seen that for n jobs there are $n!$ sequences are possible. Let σ be one of the $n!$ sequences. Let $C_{[j]}$ be the completion time of job in position [j] of the sequence. The earliness and tardiness of job in position [j] of the sequence are given as

$$E_{[j]} = \mathrm{Max}\{0, (k - C_{[j]})\} \tag{2}$$

$$T_{[j]} = \mathrm{Max}\{0, (C_{[j]} - k)\} \tag{3}$$

Cheng (1992) formulated the problem of finding the joint optimal common due date and the optimal job sequence as a constrained convex program as given below:

$$\text{Minimize} \quad f(k, \sigma) = \sum_{j=1}^{n} \{E_{[j]} + T_{[j]}\} \tag{4}$$

$$k - E_{[j]} + T_{[j]} = C_{[j]} \tag{5}$$

$$k, E_{[j]}, T_{[j]} \geqslant 0 \quad \text{for all } j = 1, \ldots, n \tag{6}$$

It is shown in earlier studies (Panwalkar et al., 1982; Cheng, 1992) that there exists an optimal common due-date k such that one of the jobs completed at time k. It is also shown in Panwalkar et al. (1982) and Cheng (1992) that there exists an optimal sequence in which the r-th job is completed at the due-date k. The value of r is given as

$$r = \frac{n+1}{2} \quad \text{if } n \text{ is odd} \tag{7}$$

$$r = \frac{n}{2} \quad \text{if } n \text{ is even} \tag{8}$$

The optimal value of the common due-date (k^*) is the sum of processing times of jobs in the first r positions in the sequence; i.e.,

$$k^* = C_{[r]} \tag{9}$$

When the learning effect is not considered, the optimal sequence of jobs is obtained by using a simple theorem in algebra by Cheng (1992). For the same problem, the optimal sequence of jobs is obtained by using a simple algorithm in Panwalkar et al. (1982). In the algorithm given in Panwalkar et al. (1982), the positional weight of position j, in the sequence $j = 1, 2, \ldots, n$, are obtained. The optimal sequence is obtained by the well-known matching procedure of the largest processing time to the smallest positional weight, the next larger processing time to the next positional weight, etc. The positional weights γ_j are given by

$$\gamma_j = (j - 1) \quad \text{if } 1 \leqslant j \leqslant r \tag{10}$$

$$\gamma_j = (n + 1 - j) \quad \text{if } (r + 1) \leqslant j \leqslant n \tag{11}$$

It is shown in Cheng (1992) that there are 2^{r-1} optimal sequences to this problem if n is odd, and 2^r optimal sequences if n is even.

Organization of this paper: This paper is organized as follows: we first consider the single machine scheduling problem with a learning effect, and present an Algorithm 1, to obtain the optimal sequence. We show that this Algorithm 1 gives an unique sequence and also show that this unique sequence is the optimal sequence only when $B_\alpha < \alpha < 0$. Next, we consider the same problem with an aging effect, and present an Algorithm 2, to obtain the optimal sequence. We show that this Algorithm 2 gives an unique sequence and also show that this unique sequence is the optimal sequence only when $0 < \alpha < A_\alpha$. We derive the bounds B_α and B_α^* for the learning effect and the bounds A_α and A_α^* for the aging effect. We also present a numerical example with seven jobs for ease of understanding and a generalized proof for any number of jobs.

3. Single machine scheduling with a learning effect

In this section, we will consider the single machine scheduling problem presented by Cheng (1992) with a learning effect. We now present the algorithm to obtain the optimal sequence of jobs.

Algorithm 1. With a learning effect

Step 1: Compute the value of r using Eq. (7) or (8).
Step 2: For each position j ($j = 1, 2, \ldots, n$), obtain the positional weights γ_j using Eqs. (10) and (11).
Step 3: Rank the positional labels/index γ_j in descending order of magnitude such that the largest γ_j is ranked 1 and the smallest γ_j is ranked n. Break the ties in the following manner.
Step 3a: Let there be a $\gamma_j = 1$ in positions 1 to r, and also there be a $\gamma_j = 1$ in positions $(r + 1)$ to n. When the γ_j is same, the γ_j in positions 1 to r should be ranked before the γ_j in positions $(r + 1)$ to n.
Step 4: Obtain the optimal sequence (S) such that job i is scheduled in position j corresponding to γ_j ranked in position i.
Step 5: Set the common due-date $k^* = p_{[1]} + p_{[2]} + \cdots + p_{[r]}$. Here $p_{[i]}$ is the processing time of job in position i.
Step 6: The objective function value is computed from Eq. (4), with this k^* and S.

Inclusion of learning effect: The learning effect is included in Algorithm 1 in Step 3a. In this Step 3a, the ties are not broken arbitrarily. Hence, the sequence we obtain at the termination of this Algorithm 1 is an unique sequence.

Conjecture 1. *We show that this unique sequence is the optimal sequence only when $B_\alpha < \alpha < 0$. We will also present a way of obtaining the bound B_α.*

Table 1
Steps of Algorithm 1 (with a learning effect).

Position-j	1	2	3	4	5	6	7
γ_j	0	1	2	3	3	2	1
Rank-i	7	5	3	1	2	4	6

Example with a learning effect: Consider the same example given in Cheng (1992), with a learning effect, to explain Algorithm 1. The normal processing times for jobs are $p_1 = 1$, $p_2 = 3$, $p_3 = 6$, $p_4 = 8$, $p_5 = 11$, $p_6 = 15$, and $p_7 = 21$. In this example $n = 7$ the number of jobs. Hence, the value of $r = 4$ using Eq. (7). We assume that $-0.6309 < \alpha < 0$.

The positional weights γ_j are obtained using Eqs. (10) and (11). The value of positional weights γ_j is 0, 1, 2, and 3. We see that there is a $\gamma_j = 1, \gamma_j = 2, \gamma_j = 3$ in positions 1 to r, and there is a $\gamma_j = 1$, $\gamma_j = 2$, $\gamma_j = 3$ in positions $(r + 1)$ to n.

In Algorithm 1, the γ_j in positions 1 to r should be ranked before the γ_j in positions $(r + 1)$ to n. In other words, the positional weights are ranked as given in Step 3a of Algorithm 1. Hence, the ranking obtained is as shown in Table 1.

Hence, for this problem, we get the unique sequence {7 5 3 1 2 4 6} at the termination of Algorithm 1. Let this unique sequence obtained at the termination of Algorithm 1 be $S^* = \{7\,5\,3\,1\,2\,4\,6\}$. The common due-date k^* for this sequence S^* is obtained from Step 5 and is given as

$$k^* = p_7 1^\alpha + p_5 2^\alpha + p_3 3^\alpha + p_1 4^\alpha \tag{12}$$

We will now prove that the unique sequence S^* obtained from our Algorithm 1 is the optimal sequence.

We can see that the sequence S^* obtained at the termination of Algorithm 1 is one of the eight sequences ($S^* = S_4$) obtained when the learning effect is not considered. In order to prove that S^* is the optimal sequence, we have to show the following:

$$f(k^*, S^*) \leqslant f(k_q, S_q) \quad \text{for all } q = 1, 2, \ldots, 8 \text{ and } q \neq 4 \tag{13}$$

The value of $f(k^*, S^*)$ is given as

$$f(k^*, S^*) = \sum_{i=1}^{n} \{E_{[i]} + T_{[i]}\}$$

$$f(k^*, S^*) = (p_5 2^\alpha + p_3 3^\alpha + p_1 4^\alpha) + (p_3 3^\alpha + p_1 4^\alpha) + p_1 4^\alpha + 0 + p_2 5^\alpha + (p_2 5^\alpha + p_4 6^\alpha) + (p_2 5^\alpha + p_4 6^\alpha + p_6 7^\alpha)$$

$$f(k^*, S^*) = p_5 2^\alpha + 2p_3 3^\alpha + 3p_1 4^\alpha + 3p_2 5^\alpha + 2p_4 6^\alpha + p_6 7^\alpha \tag{14}$$

Let us consider the sequence $S_3 = \{7\,5\,3\,2\,1\,4\,6\}$. We show that the sequence S^* is a better sequence than the sequence S_3. For this, we have to prove that $f(k^*, S^*) < f(k_3, S_3)$. Note that in sequence S^* if the jobs in positions 4 and 5 are interchanged, then we get the sequence S_3. The common due-date k_3 for this sequence S_3 is given as

$$k_3 = p_7 1^\alpha + p_5 2^\alpha + p_3 3^\alpha + p_2 4^\alpha \tag{15}$$

The value of $f(k_3, S_3)$ is given as

$$f(k_3, S_3) = \sum_{i=1}^{n} \{E_{[i]} + T_{[i]}\}$$

$$f(k_3, S_3) = (p_5 2^\alpha + p_3 3^\alpha + p_2 4^\alpha) + (p_3 3^\alpha + p_2 4^\alpha) + p_2 4^\alpha + 0 + p_1 5^\alpha + (p_1 5^\alpha + p_4 6^\alpha) + (p_1 5^\alpha + p_4 6^\alpha + p_6 7^\alpha)$$

$$f(k_3, S_3) = p_5 2^\alpha + 2p_3 3^\alpha + 3p_2 4^\alpha + 3p_1 5^\alpha + 2p_4 6^\alpha + p_6 7^\alpha \tag{16}$$

If the sequence $S_3 = \{7\,5\,3\,2\,1\,4\,6\}$ is a better sequence than the sequence $S^* = \{7\,5\,3\,1\,2\,4\,6\}$, then $f(k_3, S_3) < f(k^*, S^*)$. Let $X = f(k_3, S_3) - f(k^*, S^*)$. The value of X is obtained using Eqs. (14) and (16) as

$$X = (3p_2 4^\alpha + 3p_1 5^\alpha) - (3p_1 4^\alpha + 3p_2 5^\alpha) \tag{17}$$

If $X > 0$, then $f(k_3, S_3) > f(k^*, S^*)$, and so the sequence S^* is a better sequence than the sequence S_3. We will prove that $X > 0$ below. From Eq. (17), $X > 0$ means that

$$(3p_2 4^\alpha + 3p_1 5^\alpha) > (3p_1 4^\alpha + 3p_2 5^\alpha)$$
$$(p_2 4^\alpha + p_1 5^\alpha) > (p_1 4^\alpha + p_2 5^\alpha) \tag{18}$$

This Eq. (18) reduces to

$$4^\alpha (p_2 - p_1) > 5^\alpha (p_2 - p_1) \tag{19}$$

In Eq. (19), we know that $p_2 > p_1$. Also we know that $4^\alpha > 5^\alpha$, because α is a negative quantity. Hence, $X > 0$. This implies that $f(k_3, S_3) > f(k^*, S^*)$. Hence, the sequence S^* is a better sequence than the sequence S_3.

Note that when $\alpha = 0$, then the value of X given by Eq. (17) is 0, which implies that $f(k_2, S_2) = f(k^*, S^*)$. We can also see from Eqs. (17) and (19) that the optimal sequence is independent of the value of α, when $\alpha < 0$.

Let us consider another sequence $S_2 = \{7\,5\,4\,1\,2\,3\,6\}$. Note that in sequence S^* if the jobs in positions 3 and 6 are interchanged, then we get the sequence S_2. The common due-date k_2 for this sequence S_2 is given as

$$k^* = p_7 1^\alpha + p_5 2^\alpha + p_4 3^\alpha + p_1 4^\alpha \tag{20}$$

The value of $f(k_2, S_2)$ is given as

$$f(k_2, S_2) = \sum_{i=1}^{n} \{E_{[i]} + T_{[i]}\}$$

$$f(k_2, S_2) = (p_5 2^\alpha + p_4 3^\alpha + p_1 4^\alpha) + (p_4 3^\alpha + p_1 4^\alpha) + p_1 4^\alpha + 0 + p_2 5^\alpha + (p_2 5^\alpha + p_3 6^\alpha) + (p_2 5^\alpha + p_3 6^\alpha + p_6 7^\alpha)$$

$$f(k_2, S_2) = p_5 2^\alpha + 2p_4 3^\alpha + 3p_1 4^\alpha + 3p_2 5^\alpha + 2p_3 6^\alpha + p_6 7^\alpha \tag{21}$$

If the sequence $S_2 = \{7\,5\,4\,1\,2\,3\,6\}$ is a better sequence than the sequence $S^* = \{7\,5\,3\,1\,2\,4\,6\}$, then $f(k_2, S_2) < f(k^*, S^*)$. Let $X = f(k_2, S_2) - f(k^*, S^*)$. The value of X is obtained using Eqs. (14) and (21) as

$$X = (2p_4 3^\alpha + 2p_3 6^\alpha) - (3p_3 3^\alpha + 2p_4 6^\alpha) \tag{22}$$

If $X > 0$ then $f(k_2, S_2) > f(k^*, S^*)$, and so the sequence S^* is a better sequence than the sequence S_2. We will prove that $X > 0$ below. From Eq. (22), $X > 0$ means that

$$(2p_4 3^\alpha + 2p_3 6^\alpha) > (2p_3 3^\alpha + 2p_4 6^\alpha)$$
$$(p_4 3^\alpha + p_3 6^\alpha) > (p_3 3^\alpha + p_4 6^\alpha) \tag{23}$$

This Eq. (23) reduces to

$$3^\alpha (p_4 - p_3) > 6^\alpha (p_4 - p_3) \tag{24}$$

In Eq. (24), we know that $p_4 > p_3$. Also we know that $3^\alpha > 6^\alpha$, because α is a negative quantity. Hence, $X > 0$. This implies that $f(k_2, S_2) > f(k^*, S^*)$. Hence, here also the sequence S^* is a better sequence than the sequence S_2. In the same way, we can prove that the sequence S^* is a better sequence than the other sequences S_3, S_5, S_6, S_7 and S_8. Hence, when the learning effect is considered, the unique sequence we obtain at the termination of Algorithm 1 is the optimal sequence.

Here also, note that when $\alpha = 0$, then the value of X given by Eq. (22) is 0, which implies that $f(k_2, S_2) = f(k^*, S^*)$. So, when the learning effect is not considered i.e., $\alpha = 0$, the optimal sequence is not unique. It is important to note that this analysis is valid only when $-0.6309 < \alpha < 0$.

Numerical example: We now include the learning effect for example given in Cheng (1992). The normal processing times for jobs are $p_1 = 1$, $p_2 = 3$, $p_3 = 6$, $p_4 = 8$, $p_5 = 11$, $p_6 = 15$, and $p_7 = 21$. The processing time of jobs at various positions in the sequence is given in Table 2, for the value of $\alpha = -0.515$. Note that the value of α is in $-0.6309 < \alpha < 0$.

The value of common due-date value k_1 to k_8, and the objective function value for all the eight sequences S_1 to S_8 are given in Table 3. From this Table 3, we can see that the sequence obtained from Algorithm 1 is the optimal sequence.

4. Single machine scheduling with an aging effect

In this section, we will consider the single machine scheduling problem presented by Cheng (1992) with an aging effect. For this case (aging effect) also, the jobs are numbered according to the shortest normal processing time rule, i.e., $p_1 \leqslant p_2 \leqslant \cdots \leqslant p_n$. Because of the aging effect, the processing time of a job depends on its position in the sequence. Hence, the processing time of a job is given as

$$p_{jl} = p_j l^\alpha \tag{25}$$

Table 2
Processing time of jobs according to positions for $\alpha = -0.515$.

Job-j	Positions l						
	1	2	3	4	5	6	7
1	1	0.6988	0.5679	0.4879	0.4365	0.3974	0.3671
2	3	2.0994	1.7037	1.4691	1.3095	1.1922	1.1013
3	6	4.1988	3.4074	2.9382	2.6190	2.3844	2.2026
4	8	5.5984	4.5432	3.9176	3.4920	3.1792	2.9368
5	11	7.6978	6.2469	5.3867	4.8015	4.3714	4.0381
6	15	10.4970	8.5185	7.3455	6.5475	5.9610	5.5065
7	21	14.6958	11.9259	10.2837	9.1665	8.3454	7.7091

Table 3
Results for the numerical example with $\alpha = -0.515$.

Sequence	Common due-date	Objective function value
S_1	$k_1 = \{p_7 1^\alpha + p_5 2^\alpha + p_4 3^\alpha + p_2 4^\alpha\} = 33.5743$	$f(k_1, S_1) = 32.7768$
S_2	$k_2 = \{p_7 1^\alpha + p_5 2^\alpha + p_4 3^\alpha + p_1 4^\alpha\} = 33.7307$	$f(k_2, S_2) = 32.4578$
S_3	$k_3 = \{p_7 1^\alpha + p_5 2^\alpha + p_3 3^\alpha + p_2 4^\alpha\} = 33.5743$	$f(k_3, S_3) = 32.0949$
$S_4 = S^*$	$k^* = k_4 = \{p_7 1^\alpha + p_5 2^\alpha + p_3 3^\alpha + p_1 4^\alpha\} = 32.5949$	$f(k^*, S^*) = 31.7759$
S_5	$k_5 = \{p_7 1^\alpha + p_6 2^\alpha + p_4 3^\alpha + p_2 4^\alpha\} = 37.5093$	$f(k_5, S_5) = 34.1076$
S_6	$k_6 = \{p_7 1^\alpha + p_6 2^\alpha + p_4 3^\alpha + p_1 4^\alpha\} = 36.5299$	$f(k_6, S_6) = 33.7886$
S_7	$k_7 = \{p_7 1^\alpha + p_6 2^\alpha + p_3 3^\alpha + p_2 4^\alpha\} = 36.3735$	$f(k_7, S_7) = 33.4257$
S_8	$k_8 = \{p_7 1^\alpha + p_6 2^\alpha + p_3 3^\alpha + p_1 4^\alpha\} = 35.3941$	$f(k_6, S_6) = 33.1067$

Table 4
Steps of Algorithm 2 (with an aging effect).

Position-j	1	2	3	4	5	6	7
γ_j	0	1	2	3	3	2	1
Rank-i	7	6	4	2	1	3	5

In the above equation, p_{jl} is the processing time of job j if it is in position l of the sequence, and α here is the aging index and $\alpha > 0$. From Eq. (25), we see that $p_{j1} < p_{j2} < p_{j3} < \cdots < p_{jn}$. For example, if $p_j = 3$ and $\alpha = 0.515$, then $p_{j1} = 3$, $p_{j2} = 4.2870$, $p_{j3} = 5.2825$, $p_{j4} = 6.1261$, $p_{j5} = 6.8721$, and so on. We can see that the actual processing time of job j increases when its position in the sequence increases.

We now present an Algorithm 2, to obtain the optimal sequence of jobs.

Algorithm 2. With an aging effect

Step 1: Compute the value of r using Eq. (7) or (8).
Step 2: For each position j ($j = 1, 2, \ldots, n$), obtain the positional weights γ_j using Eqs. (10) and (11).
Step 3: Rank the positional labels/index γ_j in descending order of magnitude such that the largest γ_j is ranked 1 and the smallest γ_j is ranked n. Break the ties in the following manner.
Step 3a: Let there is a $\gamma_j = 1$ in positions 1 to r, and also there is a $\gamma_j = 1$ in positions $(r+1)$ to n. When the γ_j is same, the γ_j in positions $(r+1)$ to n should be ranked before the γ_j in positions 1 to r.
Step 4: Obtain the optimal sequence (S) such that job i is scheduled in position j corresponding to γ_j ranked in position i.
Step 5: Set the common due-date $k^* = p_{[1]} + p_{[2]} + \cdots + p_{[r]}$. Here $p_{[i]}$ is the processing time of job in position i.
Step 6: The objective function value is computed from Eq. (4), with this k^* and S.

Inclusion of aging effect: The aging effect is included in Algorithm 2 in Step 3a. In this Step 3a, the ties are not broken arbitrarily. Hence, the sequence we obtain at the termination of this Algorithm 2 is an unique sequence.

Conjecture 2. *We show that this unique sequence is the optimal sequence only when $0 < \alpha < A_\alpha$. We will also present a way of obtaining the bound A_α.*

This above conjecture can be proved in the same manner as the proof of Conjecture 1 presented in the earlier section.

Table 5
Processing time of jobs according to positions for $\alpha = 0.515$.

Job-j	Positions						
	1	2	3	4	5	6	7
1	1	1.4290	1.7608	2.0420	2.2907	2.5162	2.7241
2	3	4.2870	5.2825	6.1261	6.8721	7.5486	8.1723
3	6	8.5740	10.5650	12.2521	13.7442	15.0973	16.3447
4	8	11.4320	14.0866	16.3362	18.3257	20.1297	21.7929
5	11	15.7189	19.3691	22.4623	25.1978	27.6784	29.9653
6	15	21.4349	26.4125	30.6304	34.3606	37.7432	40.8617
7	21	30.0089	36.9774	42.8825	48.1048	52.8405	57.2064

Numerical example: We now include the aging effect for example given in Cheng (1992). The normal processing times for jobs are $p_1 = 1$, $p_2 = 3$, $p_3 = 6$, $p_4 = 8$, $p_5 = 11$, $p_6 = 15$, and $p_7 = 21$. The ranking obtained using Step 3a of Algorithm 2 is shown in Table 4. The processing time of jobs at various positions in the sequence is given in Table 5, for the value of $\alpha = 0.515$. Note that $0 < \alpha < 0.8181$.

The value of common due-date value k_1 to k_8, and the objective function value for all the eight sequences S_1 to S_8 are given in Table 6. From this Table 6, we can see that the sequence obtained from Algorithm 3 is the optimal sequence.

5. Generalization with a learning/aging effect

In this section, we will give the proof of optimal sequence for any number of jobs. Here, we assume that for learning effect $B_\alpha < \alpha < 0$ and for aging effect $0 < \alpha < A_\alpha$.

Inclusion of learning effect: We consider n the number of jobs is odd. Using Eq. (7), we obtain $r = (n+1)/2$. The positional weights γ_j can be obtained using Eqs. (10) and (11). The value of positional weights γ_j ($j = 1, 2, \ldots, n$) vary from 0 to $r-1$. The value of $\gamma_1 = 0$; the value of γ_2 and γ_n are 1; the value of γ_3 and γ_{n-1} are 2. In the same way the value of γ_r and γ_{r+1} are $r-1$.

In Algorithm 1 (Step 3a), the γ_j in positions 1 to r should be ranked before the γ_j in positions $(r+1)$ to n. We obtain the following unique sequence at the termination of Algorithm 1. Let this unique sequence obtained at the termination of Algorithm 1 be S^* and is

$$S^* = \{n, (n-2), (n-4), \ldots 1, 2, \ldots (n-5), (n-3), (n-1)\} \tag{26}$$

Table 6
Results for the numerical example with $\alpha = 0.515$.

Sequence	Common due-date	Objective function value
S_1	$k_1 = \{p_7 1^\alpha + p_5 2^\alpha + p_4 3^\alpha + p_2 4^\alpha\} = 56.9316$	$f(k_1, S_1) = 140.1989$
S_2	$k_2 = \{p_7 1^\alpha + p_5 2^\alpha + p_4 3^\alpha + p_1 4^\alpha\} = 52.8476$	$f(k_2, S_2) = 141.6910$
S_3	$k_3 = \{p_7 1^\alpha + p_5 2^\alpha + p_3 3^\alpha + p_2 4^\alpha\} = 53.4100$	$f(k_3, S_3) = 143.2204$
S_4	$k_4 = \{p_7 1^\alpha + p_5 2^\alpha + p_3 3^\alpha + p_1 4^\alpha\} = 49.3259$	$f(k_4, S_4) = 144.7125$
$S_5 = S^*$	$k^* = k_5 = \{p_7 1^\alpha + p_6 2^\alpha + p_4 3^\alpha + p_2 4^\alpha\} = 62.6476$	$f(k^*, S^*) = 135.0184$
S_6	$k_6 = \{p_7 1^\alpha + p_6 2^\alpha + p_4 3^\alpha + p_1 4^\alpha\} = 58.5636$	$f(k_6, S_6) = 136.5105$
S_7	$k_7 = \{p_7 1^\alpha + p_6 2^\alpha + p_3 3^\alpha + p_2 4^\alpha\} = 59.1260$	$f(k_7, S_7) = 138.0399$
S_8	$k_8 = \{p_7 1^\alpha + p_6 2^\alpha + p_3 3^\alpha + p_1 4^\alpha\} = 55.0419$	$f(k_6, S_6) = 139.5320$

The common due-date k^* for this sequence S^* is obtained from Step 5 and is

$$k^* = p_n 1^\alpha + p_{(n-2)} 2^\alpha + p_{(n-4)} 3^\alpha + \cdots + p_1 r^\alpha \tag{27}$$

We will now prove that the unique sequence S^* obtained from our Algorithm 1 is the optimal sequence. The value of $f(k^*, S^*)$ is given as

$$f(k^*, S^*) = \sum_{i=1}^{n} \{E_{[i]} + T_{[i]}\}$$

$$\begin{aligned} f(k^*, S^*) = {} & p_{(n-2)} 2^\alpha + 2p_{(n-4)} 3^\alpha + 3p_{(n-6)} 4^\alpha \\ & + \cdots + (r-1)p_1 r^\alpha + (r-1)p_2 (r+1)^\alpha \\ & + \cdots + 3p_{(n-5)}(n-2)^\alpha + 2p_{(n-3)}(n-1)^\alpha \\ & + p_{(n-1)} n^\alpha \end{aligned} \tag{28}$$

We can also see that the sequence S^* obtained at the termination of Algorithm 1 is one of the 2^{r-1} sequences obtained when the learning effect is not considered.

Proof. We prove that S^* is the optimal sequence by considering any one of the other 2^{r-1} sequences obtained when the learning effect is not considered.

Let us consider the sequence S_a given as

$$S_a = \{n, (n-2), (n-4), \ldots 2, 1, \ldots (n-5), (n-3), (n-1)\} \tag{29}$$

We show that the sequence S^* is a better sequence than the sequence S_a. For this, we have to prove that $f(k^*, S^*) < f(k_a, S_a)$. Note that in sequence S^* if the jobs in positions r and $r+1$ are interchanged, then we get the sequence S_a. The common due-date k_a for this sequence S_a is given as

$$k_a = p_n 1^\alpha + p_{(n-2)} 2^\alpha + p_{(n-4)} 3^\alpha + \cdots + p_2 r^\alpha \tag{30}$$

The value of $f(k_a, S_a)$ is given as

$$f(k_a, S_a) = \sum_{i=1}^{n} \{E_{[i]} + T_{[i]}\}$$

$$\begin{aligned} f(k_a, S_a) = {} & p_{(n-2)} 2^\alpha + 2p_{(n-4)} 3^\alpha + 3p_{(n-6)} 4^\alpha \\ & + \cdots + (r-1)p_2 r^\alpha + (r-1)p_1 (r+1)^\alpha \\ & + \cdots + 3p_{(n-5)}(n-2)^\alpha + 2p_{(n-3)}(n-1)^\alpha \\ & + p_{(n-1)} n^\alpha \end{aligned} \tag{31}$$

If sequence S_a is a better sequence than the sequence S^*, then $f(k_a, S_a) < f(k^*, S^*)$. Let $X = f(k_a, S_a) - f(k^*, S^*)$ and is

$$\begin{aligned} X = {} & \{(r-1) * p_2 r^\alpha + (r-1) * p_1 (r+1)^\alpha\} \\ & - \{(r-1) * p_1 r^\alpha + (r-1) * p_2 (r+1)^\alpha\} \end{aligned} \tag{32}$$

If $X > 0$, then $f(k_a, S_a) > f(k^*, S^*)$, and so the sequence S^* is a better sequence than the sequence S_a. We will prove below that $X > 0$. From Eq. (32), $X > 0$ means that

$$\begin{aligned} & \{(r-1)p_2 r^\alpha + (r-1)p_1 (r+1)^\alpha\} \\ & \quad > \{(r-1)p_1 r^\alpha + (r-1)p_2 (r+1)^\alpha\} \\ & \{p_2 r^\alpha + p_1 (r+1)^\alpha\} > \{p_1 r^\alpha + p_2 (r+1)^\alpha\} \end{aligned} \tag{33}$$

This Eq. (33) reduces to

$$r^\alpha (p_2 - p_1) > (r+1)^\alpha (p_2 - p_1) \tag{34}$$

In Eq. (34), we know that $p_2 > p_1$. Also we know that $r^\alpha > (r+1)^\alpha$, because α is a negative quantity. Hence, $X > 0$. This implies that $f(k_a, S_a) > f(k^*, S^*)$. Hence, the sequence S^* is a better sequence than the sequence S_a. When $\alpha = 0$, then the value of X given by Eq. (32) is 0, which implies that $f(k_a, S_a) = f(k^*, S^*)$.

In the same manner, we can prove that for any sequence S_q, $f(k_q, S_q) > f(k^*, S^*)$ and hence S^* is the unique optimal sequence. Note that in our proof we have taken n is odd. A similar proof can be easily obtained when n is even. It is important to note that this analysis is valid only when $B_\alpha < \alpha < 0$.

Inclusion of aging effect: We consider n the number of jobs is odd. Using Eq. (7), we obtain $r = (n+1)/2$. The positional weights γ_j can be obtained using Eqs. (10) and (11). The value of positional weights γ_j $(j = 1, 2, \ldots, n)$ vary from 0 to $r-1$. The value of $\gamma_1 = 0$; the value of γ_2 and γ_n are 1; the value of γ_3 and γ_{n-1} are 2. In the same way the value of γ_r and γ_{r+1} are $r-1$.

In Algorithm 2 (Step 3a), the γ_j in positions $(r+1)$ to n should be ranked before the γ_j in positions 1 to r. We obtain the following unique sequence at the termination of Algorithm 2. Let this unique sequence obtained at the termination of Algorithm 2 be S^* and is

$$S^* = \{n, (n-1), (n-3), \ldots 2, 1, \ldots (n-6), (n-4), (n-2)\} \tag{35}$$

The common due-date k^* for this sequence S^* is obtained from Step 5 and is

$$k^* = p_n 1^\alpha + p_{(n-1)} 2^\alpha + p_{(n-3)} 3^\alpha + \cdots + p_1 r^\alpha \tag{36}$$

We can easily prove that the unique sequence S^* obtained from our Algorithm 2 is the optimal sequence, in the same manner as done before with learning effect.

6. Derivation of bounds on learning/aging effect

In the earlier section, we have shown that Algorithm 1 will give unique optimal sequence when $B_\alpha<\alpha<0$ for learning effect. We have also shown that Algorithm 2 will give unique optimal sequence when $0<\alpha<A_\alpha$ for aging effect. In this section, we will show how to obtain these bounds.

Bounds for learning effect: Let n be odd. The bounds are obtained using the positional weights obtained from (10) and (11). The positional weights γ_2 and γ_n are same and is 1. In Algorithm 1, position n is assigned before position 2. This means that $\gamma_2 * 2^\alpha > \gamma_n * n^\alpha$ which is true because $\alpha<0$.

The positional weights γ_2 and γ_{n-1} are 1 and 2, respectively. In Algorithm 1, position 2 is assigned before position $(n-1)$. This means that $\gamma_2 * 2^\alpha > \gamma_{n-1} * (n-1)^\alpha$. If $\gamma_2 * 2^\alpha = \gamma_{n-1} * (n-1)^\alpha$, there are two sequences that are optimal. So we have to find the maximum value of α (given as B_α) for which $\gamma_2 * 2^\alpha > \gamma_{n-1} * (n-1)^\alpha$. This can be obtained by equating $\gamma_2 * 2^\alpha$ and $\gamma_{n-1} * (n-1)^\alpha$ and is

$$B_\alpha = \frac{\log(2) - \log(1)}{\log(2) - \log(n-1)} \tag{37}$$

The positional weights γ_2 and γ_3 are also 1 and 2, respectively. In Algorithm 1, position 2 is assigned before position 3. This means that $\gamma_2 * 2^\alpha > \gamma_3 * 3^\alpha$. If $\gamma_2 * 2^\alpha = \gamma_3 * 3^\alpha$, there are two sequences that are optimal. So we have to find the maximum value of α (given as B^*_α) for which $\gamma_2 * 2^\alpha > \gamma_3 * 3^\alpha$. This can be obtained by equating $\gamma_2 * 2^\alpha$ and $\gamma_3 * 3^\alpha$ and is

$$B^*_\alpha = \frac{\log(2) - \log(1)}{\log(2) - \log(3)} \tag{38}$$

Hence, Algorithm 1 gives unique optimal sequence only when $B_\alpha<\alpha<0$.

When $\alpha<B^*_\alpha$, we can see that $\gamma_1 = 0$, $\gamma_2 * 2^\alpha > \gamma_3 * 3^\alpha > \cdots > \gamma_n * n^\alpha$. Hence, the optimal sequence obtained by arranging the longest job in first position and the rest of the jobs in SPT order.

When $B^*_\alpha<\alpha<B_\alpha$, the learning effect can be incorporated in the positional weights as

$$\gamma_j = (j-1) * j^\alpha \quad \text{if } 1 \leqslant j \leqslant r \tag{39}$$

$$\gamma_j = (n+1-j) * j^\alpha \quad \text{if } (r+1) \leqslant j \leqslant n \tag{40}$$

The optimal sequence can be obtained by using the positional weights given by (39) and (40) (Panwalkar et al., 1982).

Bounds for aging effect: Let n be odd. The bounds are obtained using the positional weights obtained from (10) and (11). The positional weights γ_2 and γ_n are same and is 1. In Algorithm 2, position 2 is assigned before position n. This means that $\gamma_n * n^\alpha > \gamma_2 * 2^\alpha$ which is true because $\alpha>0$.

The positional weights γ_3 and γ_n are 2 and 1, respectively. In Algorithm 2, position n is assigned before position 3. This means that $\gamma_n * n^\alpha > \gamma_3 * 3^\alpha$. If $\gamma_n * n^\alpha = \gamma_3 * 3^\alpha$, there are two sequences that are optimal. So we have to find the minimum value of α (given as A_α) for which $\gamma_n * n^\alpha > \gamma_3 * 3^\alpha$. This can be obtained by equating $\gamma_n * n^\alpha$ and $\gamma_3 * 3^\alpha$ and is

$$A_\alpha = \frac{\log(2) - \log(1)}{\log(n) - \log(3)} \tag{41}$$

The positional weights γ_n and $\gamma_{(n-1)}$ are also 1 and 2, respectively. In Algorithm 2, position n is assigned before position $(n-1)$. This means that $\gamma_n * n^\alpha > \gamma_{(n-1)} * (n-1)^\alpha$. If $\gamma_n * n^\alpha = \gamma_{(n-1)} * (n-1)^\alpha$, there are two sequences that are optimal. So we have to find the minimum value of α (given as A^*_α) for which $\gamma_n * n^\alpha > \gamma_{(n-1)} * (n-1)^\alpha$. This can be obtained by equating $\gamma_n * n^\alpha$ and $\gamma_{(n-1)} * (n-1)^\alpha$ and is

$$A^*_\alpha = \frac{\log(2) - \log(1)}{\log(n) - \log(n-1)} \tag{42}$$

Hence, Algorithm 2 gives unique optimal sequence only when $0<\alpha<A_\alpha$.

When $\alpha>A^*_\alpha$, we can see that $\gamma_1 = 0$, $\gamma_2 * 2^\alpha < \gamma_3 * 3^\alpha < \cdots < \gamma_n * n^\alpha$. Hence, the optimal sequence obtained by arranging the jobs in LPT order.

When $A_\alpha<\alpha<A^*_\alpha$, the learning effect can be incorporated in the positional weights as given in (39) and (40) and the optimal sequence can be obtained from Panwalkar et al. (1982).

We can see for the numerical example with seven jobs $B_\alpha = -0.6309$, $B^*_\alpha = -1.7095$, $A_\alpha = 0.8181$, and $A^*_\alpha = 4.4966$.

7. Conclusions

The learning/aging effect in an n job single machine scheduling problem with common due date is considered. The objective is to determine the optimal common due date and the optimal sequence of jobs that minimizes a cost function in the presence of learning/aging effect. The cost function depends on the individual job earliness and tardiness values. In this paper, two bounds B_α and B^*_α for the learning index α are derived. We have shown that when $B_\alpha<\alpha<0$, then the optimal sequence is unique and provide an $O(n \log n)$ algorithm to obtain this unique optimal sequence and the optimal common due date. We have also shown that when $\alpha<B^*_\alpha$, the optimal sequence is obtained by arranging the longest job in first position and the rest of the jobs in SPT order. Similarly, we show two bounds A_α and A^*_α for the aging index α are derived. We have shown that when $0<\alpha<A_\alpha$, then the optimal sequence is unique and provide an $O(n \log n)$ algorithm to obtain this unique optimal sequence and the optimal common due date. We also show that when $\alpha>A^*_\alpha$, the optimal sequence is obtained by arranging the jobs in LPT order. A numerical example is presented for illustration of our analysis.

Acknowledgements

The authors would like to thank one of the reviewers for his/her suggestions which were useful in deriving the bounds on learning/aging index. The authors would like to thank Editor-in-Chief T.C.E. Cheng for his encouragement.

References

Bachman, A., Janiak, A., 2004. Scheduling jobs with position-dependent processing times. 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., Simons, D., 2004. Common due date scheduling with autonomous and induced learning. European Journal of Operational Research 159, 606–616.

Cheng, T.C.E., 1984. Optimal due-date determination and sequencing of *n* jobs on a single machine. Journal of the Operational Research Society 35, 433–437.

Cheng, T.C.E., 1987. Optimal total-work-content due-date determination and sequencing. Computers and Mathematics with Applications 14, 579–582.

Cheng, T.C.E., 1991. Optimal assignment of slack due-date and sequencing of jobs with random processing times on a single machine. European Journal of Operational Research 51, 348–353.

Cheng, T.C.E., 1992. Optimal single machine sequencing and assignment of common due-dates. Computers and Industrial Engineering 22, 115–120.

Cheng, T.C.E., Gupta, M., 1989. Survey of scheduling research involving due-date determination decisions. European Journal of Operational Research 38, 156–166.

Cheng, T.C.E., Oguz, C., Qi, X.D., 1996. Due-date assignment and single machine scheduling with compressible processing times. International Journal of Production Economics 43, 29–35.

Cheng, T.C.E., Ding, Q., Kovalyov, M.Y., Bachman, A., Janiak, A., 2003. Scheduling jobs with linearly decreasing processing times. Naval Research Logistics 50, 531–555.

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.

Du, J., Leung, J.Y.T., 1990. Minimizing total tardiness on one machine is NP-hard. Mathematics of Operations Research 15, 483–495.

Kanet, J.J., 1981. Minimizing the average deviation of job completion times about a common due-date. Naval Research Logistics Quarterly 28, 634–651.

Kuo, W.H., Yang, D.L., 2006. Minimizing the makespan in a single machine scheduling problem with a time-based learning effect. Information Processing Letters 97, 64–67.

Lee, W.C., Wu, C.C., 2004. Minimizing total completion time in a two-machine flowshop with a learning effect. International Journal of Production Economics 88, 85–93.

Mosheiov, G., 2001a. Scheduling problems with a learning effect. European Journal of Operational Research 132, 687–693.

Mosheiov, G., 2001b. Parallel machine scheduling with a learning effect. Journal of the Operational Research Society 52, 1165–1169.

Mosheiov, G., Sidney, J.B., 2005. Note on scheduling with general learning curves to minimize the number of tardy jobs. Journal of the Operational Research Society 56, 110–112.

Panwalkar, S.S., Smith, M.L., Seidmann, A., 1982. Common due-date assignment to minimize total penalty for the one machine scheduling problem. Operation Research 30, 391–399.

Wang, J.B., 2006. A note on scheduling problems with learning effect and deteriorating jobs. International Journal of Systems Science 37, 827–833.

Wang, J.B., 2007. Single machine scheduling problems with the effects of learning and deterioration. Omega 354, 397–402.

Wang, J.B., Cheng, T.C.E., 2007. Scheduling problems with the effects of deterioration and learning. Asia-Pacific Journal of Operational Research 24, 245–261.

Wang, J.B., Xia, Z.Q., 2005. Flowshop scheduling with a learning effect. Journal of the Operational Research Society 56, 1325–1330.

Wang, J.B., Ng, C.T., Cheng, T.C.E., Liu, L.L., 2008. Single machine scheduling with a time-dependent learning effect. International Journal of Production Economics 111, 802–811.