

Single machine scheduling with past-sequence-dependent setup times and learning effects: A parametric analysis

V. Mani¹, Pei Chann Chang², and Shih Hsin Chen³

¹Department of Aerospace Engineering,
Indian Institute of Science, Bangalore, India.

²Department of Information Management
Yuan Ze University, 135, Yuan-Dong Road,
Tao-Yuan 320, Taiwan, R.O.C.

³Department of Electronic Commerce Management,
Nanhua University, Chiayi 62248, Taiwan, R.O.C.

Address for correspondence

Prof. Pei Chann Chang

Department of Information Management

Yuan Ze University

135, Yuan-Tung Road

Chung-Li, Tao-Yuan 320

Taiwan, R.O.C

Tel: 886-3-4636165

Fax: 886-3-4635319

Email: iepchang@saturn.yzu.edu.tw

Abstract In this paper, we consider the single machine scheduling problem with past-sequence-dependent setup times and a learning effect. The setup times are proportional to the length of jobs that are already scheduled; i.e., past-sequence-dependent (p-s-d) setup times. The learning effect reduces the actual processing time of a job because the workers are involved in doing the same job or activity repeatedly. Hence, the processing time of a job depends on its position in the sequence. In this study, we consider the total absolute difference in completion times (TADC) as the objective function. This problem is denoted as $1/LE, s_{psd}/TADC$ in [9]. There are two parameters a and b denoted constant learning index and normalizing index, respectively. In this paper, we present a parametric analysis of b on the $1/LE, s_{psd}/TADC$ problem for a given value of a . A computational algorithm is developed to obtain the number of optimal sequences and the range of b in which each of the sequences is optimal, for a given value of a . We derive two bounds b^* for the normalizing constant b and a^* for the learning index a . We also show that when $a < a^*$ or $b > b^*$, the optimal sequence is obtained by arranging the longest job in the first position and the rest of the jobs in SPT order.

Keywords Scheduling, Setup times, learning effect

1. INTRODUCTION

In a recent study, Koulamas and Kyparisis [8] introduced the concept of past-sequence-dependent setup times in single machine scheduling problems. The processing time of a job is a variable and depends on a function of its starting time in many real-world scheduling problems [1]. In fact the study [8] is the first to consider the past-sequence-dependent setup times; i.e., the setup time that is dependent on the jobs that are already scheduled. In a production environment, the workers are involved in doing the same type of job/activity on the same machine. Hence, it is possible for the workers to learn and improve their performance. So the processing time of a job reduces due to the learning. Biskup [3] is the first to address the effect of learning in the context of single machine scheduling problems. It is shown by Biskup [3] that this problem can be solved in polynomial time if the objectives are minimization of deviation from a common due-date and the sum of flow times. The learning effect on a single and parallel identical machines with the objective of minimizing the flow-time are considered in [11, 12]. The learning effect in a two machine flowshop scheduling with the objective of finding the sequence of jobs that minimizes the total completion time is given by Lee and Wu [10]. In [10], a branch and bound technique is presented. A heuristic algorithm is also presented in [10] to improve the efficiency of the branch and bound technique. Cheng and Wang [5] considers the learning effect on the processing time of jobs using a volume dependent processing time function model. Wang [14] presents when deterioration and learning effect to job processing times are involved, some single machine problems are remain polynomially solvable. Cheng et al [6] presents a concise survey of scheduling with time dependent processing times. In a recent study, Biskup [4] presents a complete discussion on why and when the learning effects might occur and a concise review of literature on scheduling with learning effects.

In this paper, we consider the non-preemptive single machine scheduling problems with past-sequence-dependent setup times along with learning effect. To the best of our knowledge, Kuo and Yang [9] is the first to study the concept of past-sequence-dependent setup times along with learning effect in single-machine scheduling problems. The objectives considered by Kuo and Yang [9] are minimizing the maximum completion time (C_{max}), total completion time (TC), total absolute difference in completion times ($TADC$), and the unit earliness, tardiness and due date penalty ($ETCP$). These scheduling problems with past-sequence-dependent setup times along with learning effect are denoted in [9] as:

Problem.(i): $1/LE, s_{psd}/C_{max}$

Problem.(ii): $1/LE, s_{psd}/TC$

Problem.(iii): $1/LE, s_{psd}/TADC$

Problem.(iv): $1/LE, s_{psd}/ETCP$.

The scheduling problem is defined in the following manner. 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. All the jobs are available at time zero. Each job has a normal processing time p_r , ($r = 1, 2, \dots, n$). The processing time of a job after learning and occupying position r in the sequence is given by

$$p_{[r]}^l = p_{[r]}r^a, \quad n = 1, 2, \dots, n \quad (1)$$

where $a \leq 0$ is a constant learning index. Let $s_{[r]}$ be the setup time of a job occupying position r in the sequence, and $s_{[r]}$ is defined as

$$s_{[1]} = 0$$

$$s_{[r]} = b \sum_{j=1}^{r-1} p_{[j]}^l \quad r = 2, 3, \dots, n \quad (2)$$

where $b \geq 0$ is a normalizing constant. In the above Eq. (2), the actual length of the setup time depends on the value of b and learning index a . Let C_r denote the completion time of job r in a sequence. It is shown in [9] that the well-known shortest processing time (SPT) sequence is optimal for both the problems Problem.(i) $(1/LE, s_{psd}/C_{max})$ and Problem.(ii) $(1/LE, s_{psd}/TC)$.

Contributions of this paper: We consider the problem $(1/LE, s_{psd}/TADC)$. For this problem, the optimal sequence depends on the value of b and learning index a . We present a parametric analysis of b on the $1/LE, s_{psd}/TADC$ problem for a given value of a . We present a computational algorithm to obtain the optimal sequence and the range of b in which each of the sequences is optimal, for a given value of a . We derive two bounds b^* for the normalizing constant b and a^* for the learning index a . We also show that when $a < a^*$ or $b > b^*$, the optimal sequence is obtained by arranging the longest job in first position and the rest of the jobs in SPT order.

In terms of the contribution for the industry, Koulamas and Kyparisis [8] indicated that the consideration of past-sequence-dependent setup times stems from high-tech manufacturing in which a batch of jobs consists of a group of electronic components mounted together on an IC board. In addition, Uzsoy et al. [13] mentioned more general manufacturing environment in which either long setup times are common. As a result, the problem is important and practical in industry.

2. Problem definition $1/LE, s_{psd}/TADC$:

In this section, we consider the single-machine scheduling problem with the objective of minimizing the total absolute difference in completion times (TADC).

The $TADC$ of the $1/LE, s_{psd}/TADC$ scheduling problem given in [9] is

$$\begin{aligned}
TADC &= \sum_{i=1}^n \sum_{j=i}^n |C_j - C_i| \\
&= \sum_{r=1}^n (r-1)(n-r+1) (s_{[r]} + p_{[r]}^l) \\
&= \sum_{r=1}^n \left\{ (r-1)(n-r+1) + b \sum_{j=r+1}^n (j-1)(n-j+1) \right\} r^a p_{[r]} \quad (3)
\end{aligned}$$

As mentioned in [9], the above Eq. (3) can be viewed as the scalar product of two vectors. One vector is $p_{[r]}$ that is the vector of processing time of jobs. The other vector is v_r that is known as positional weights vector and it is given as

$$v_r = \left\{ (r-1)(n-r+1) + b * \sum_{j=r+1}^n (j-1)(n-j+1) \right\} r^a, \quad r = 2, 3, \dots, n \quad (4)$$

In the above positional weights vector Eq. (4), the value of $v_1 = 0$ because $s_{[1]}$ is zero ($\because s_{[r]} = b \sum_{j=1}^{r-1} P_{[j]}^A$ and v_1 is an initial weight, see also [9]). It is well-known from [7] that Eq. (3) is minimized by arranging the vectors v_r and $p_{[r]}$ in opposite orders. This is also given in [9] as Lemma.1. Hence, for a given value of b and a learning index a , the optimal sequence for the $1/s_{psd}/TADC$ problem can be obtained in $O(n \log n)$ time. It can be seen that the optimal sequence depends on the values of both b and a .

Parametric analysis of b : The optimal sequence for the $1/LE, s_{psd}/TADC$ problem depends on the value of b for a given learning index a . Our interest in this study is to find the range of b and the corresponding optimal sequence for a given learning index a . The positional weight vector given by Eq. (4), plays an important role in obtaining the optimal sequence. Hence, it is important to study the variation of the positional weights with respect to b , to obtain the sequence. We first present a motivating numerical example for understanding the contributions of this paper.

Motivating Numerical Example: Let us consider the 7 job example given

in [2]. The processing time of the jobs are: $p_1 = 2$, $p_2 = 3$, $p_3 = 6$, $p_4 = 9$, $p_5 = 21$, $p_6 = 65$ and $p_7 = 82$. Let us consider the value of $a = -0.152$ proposed in [2]. For this numerical example the positional weights are:

$$\begin{aligned}
v_1 &= 0 * 1^a &= 0.0000 \\
v_2 &= (6 + 50 * b) * 2^a &= 5.4000 + 45.0000 * b \\
v_3 &= (10 + 40 * b) * 3^a &= 8.4620 + 33.8484 * b \\
v_4 &= (12 + 28 * b) * 4^a &= 9.7200 + 22.6801 * b \\
v_5 &= (12 + 16 * b) * 5^a &= 9.3959 + 12.5278 * b \\
v_6 &= (10 + 6 * b) * 6^a &= 7.6159 + 4.5695 * b \\
v_7 &= 6 * 7^a &= 4.4637
\end{aligned} \tag{5}$$

In order to study the effect of b on the optimal sequence, we plot the above values of positional weights v_r , ($r = 1, 2, \dots, n$), with the value of b . The variations of v_r , ($r = 1, 2, \dots, n$) for b values in the range of $(0, 0.5)$ is shown in Figure.1. For a given value of a , the variation of v_r with b are linear and so we call them as lines v_1, v_2, \dots, v_7 . We see that lines $v_1 = 0$ and $v_7 = 4.4637$ are independent of b . We also see that v_1 and v_7 are less than v_2, v_3, v_4, v_5 and v_6 for $b > 0$.

In Figure 1, we see that there is a range of b in which the lines v_2, v_3, v_4, v_5 and v_6 will not intersect each other. This implies that the sequence will be the same in this range. For example when $b = 0.2$, the values of $v_1 = 0$, $v_2 = 14.4$, $v_3 = 18.1117$, $v_4 = 14.2560$, $v_5 = 11.9015$, $v_6 = 8.5298$ and $v_7 = 4.4637$. The optimal sequence obtained using [7] is $\{7, 2, 1, 3, 4, 5, 6\}$. When $b = 0.25$, the values of $v_1 = 0$, $v_2 = 16.65$, $v_3 = 16.9241$, $v_4 = 15.3900$, $v_5 = 12.5278$, $v_6 = 8.7583$ and $v_7 = 4.4637$. The optimal sequence obtained using [7] is the same. This implies that in this range of b (0.2 to 0.25) the optimal sequence is unique. Also, note that in this range of b (0.2 to 0.25) all the values of v_r for $r = 2, 3, \dots, 6$ are increasing with b .

Figure 1: Variation of v_r as a function of b

Let any two lines of v_r (v_2, v_3, v_4, v_5 and v_6) intersect at some values of $b = \hat{b}$. We can see that the optimal sequence obtained when $b < \hat{b}$ is different from the optimal sequence obtained when $b > \hat{b}$. When $b = \hat{b}$, we have two optimal sequences. Hence, in order to obtain the range of b in which a sequence is optimal, we have to obtain the intersection points of all lines v_r (v_2, v_3, v_4, v_5 and v_6) for $b > 0$.

We can obtain the intersection points by equating the positional weights v_r , ($r = 1, 2, \dots, n$) given by Eq. (5). For example ($a = -0.152$), the point of intersection of lines v_2 and v_3 is obtained as: $v_2 = v_3$, which implies $5.4000 + 45.0000 * b = 8.4620 + 33.8484 * b$. From this we get $11.152 * b = 3.062$ and hence $b = 0.2746$. There are six points of intersection for this example ($n = 7$) and are denoted as m_1 to m_6 in Figure 1. These intersection points are: Lines v_2 and v_3 will intersect at point $m_1 = 0.2746$. Lines v_2 and v_4 will intersect at point $m_2 = 0.1935$. Lines v_2 and v_5 will intersect at point $m_3 = 0.1231$. Lines v_2 and v_6 will intersect at point $m_4 = 0.0548$. Lines v_3

and v_4 will intersect at point $m_5 = 0.1126$. Lines v_3 and v_5 will intersect at point $m_6 = 0.0438$.

We arrange these 6 intersection points m_1 to m_6 in the increasing order given as $\{m_6, m_4, m_5, m_3, m_2, m_1\}$. We choose a value b in between any two consecutive values of m (say between m_4 and m_5) and obtain the optimal sequence using [7]. This sequence is optimal in the range of b given by m_4 and m_5 . In this manner, we obtain 7 optimal sequences. The optimal sequences and the range of b are shown in Table.1. Note that, we have to use one value of b in the range $0 < b < m_6$ and another value of b in the range $b > m_1$ and obtain their corresponding optimal sequences.

Table 1: Range of b and the optimal sequence for $1/LE, s_{psd}/TADC$ problem ($a = -0.152$)

Range of b	Optimal Sequence
$0 < b < m_6$	{7, 5, 3, 1, 2, 4, 6}
$m_6 < b < m_4$	{7, 5, 2, 1, 3, 4, 6}
$m_4 < b < m_5$	{7, 4, 2, 1, 3, 5, 6}
$m_5 < b < m_3$	{7, 4, 1, 2, 3, 5, 6}
$m_3 < b < m_2$	{7, 3, 1, 2, 4, 5, 6}
$m_2 < b < m_1$	{7, 2, 1, 3, 4, 5, 6}
$b > m_1$	{7, 1, 2, 3, 4, 5, 6}

From the above numerical example, we observe the following: The longest job (job number 7) will always occupy the first position in the optimal sequence (because $v_1 = 0$). The second longest job will always occupy the last position in the optimal sequence (because $v_6 < v_2, v_3, v_4, v_5$ for $b > 0$). The number of intersection points is 6. The number of optimal sequences is equal to the number of intersections plus one i.e., 7. This because we have to include the value of b for $0 < b < m_6$ and $b > m_1$. At any point of intersection there are two sequences that are optimal. For

example, when $b = 0.2746$ both the sequences $\{7, 2, 1, 3, 4, 5, 6\}$ and $\{7, 1, 2, 3, 4, 5, 6\}$ are optimal, which implies that the value of $TADC$ is same for both the sequences. For the value of $b > 0.2746$, there are no intersections of the lines. This implies that when $b > 0.2746$, the optimal sequence is unique and is $\{7, 1, 2, 3, 4, 5, 6\}$.

3. A computational algorithm for n jobs

In this section, we present a computational algorithm to obtain the optimal sequence and the range of b in which each of the sequences is optimal, for a given value of learning index a . For a general n jobs, we need to obtain the intersection points of the positional weights v_r , ($r = 1, 2, \dots, n$) for $b > 0$. The intersection points give the range of b . Once the intersection points are obtained, the optimal sequence is obtained using [7]. The computational algorithm is given below.

STEP 1. GIVEN: n the number of jobs, a the value of learning index, and m the index counter from zero.

STEP 2.

$$B(n) \leftarrow 0$$

for $r = 2$ to $n - 1$ **do**

$$A(r) = (r - 1)(n - r + 1) * r^a$$

$$B(r) = \{\sum_{j=r+1}^n (j - 1)(n - j + 1)\} * r^a$$

end for

STEP 3.

for $I = 2$ to $n - 1$ **do**

for $J = I + 1$ to $n - 1$ **do**

$$x = \frac{A(J) - A(I)}{B(I) - B(J)}$$

if $x = 0$ **then**

Do nothing

else

$Y(m) = x$ and $m = m + 1$

end if

end for

end for

STEP 4. Arrange the intersection points given by $Y(m)$ in increasing order. Let $YY(m)$ be the vector that is obtained by arranging the intersection points ($Y(m)$) in increasing order. Let b_{min} be the minimum value of $YY(m)$ and b_{max} be the maximum value of $YY(m)$.

STEP 5. Choose a value of b in between any consecutive values in $YY(m)$. With this b value, first compute the weights v_r . The optimal sequence can be obtained by arranging the elements of v_r and $p_{[r]}$ vectors in opposite order [7]. Choose one value of b in the range $0 < b < b_{min}$ and obtain the optimal sequence using [7]. Also choose one value of b in the range $b > b_{max}$ and obtain the optimal sequence in same manner using [7].

This above algorithm will give all the optimal sequences and the range of b in which each sequence is optimal, for a given value of a .

Derivation of Bounds: We also see from the values of v_r that the maximum value of b denoted as b_{max} is given by the intersection of lines v_2 and v_3 . This b_{max} value is obtained as follows: We know that

$$\begin{aligned}
 v_2 &= \left\{ (n-1) + b * \sum_{j=r+1}^n (j-1)(n-j+1) \right\} * 2^a \\
 v_3 &= \left\{ 2 * (n-2) + b * \sum_{j=r+1}^n (j-1)(n-j+1) \right\} * 3^a
 \end{aligned} \tag{6}$$

The intersection point of lines v_2 and v_3 is

$$\begin{aligned} & \left\{ (n-1) + b * \sum_{j=r+1}^n (j-1)(n-j+1) \right\} * 2^a \\ &= \left\{ 2 * (n-2) + b * \sum_{j=r+1}^n (j-1)(n-j+1) \right\} * 3^a \end{aligned}$$

This reduces to

$$\begin{aligned} b * \left\{ 2^a * \sum_{j=r+1}^n (j-1)(n-j+1) - 3^a * \sum_{j=r+1}^n (j-1)(n-j+1) \right\} \\ = \{ 2 * (n-2) * 3^a - (n-1) * 2^a \} \end{aligned} \quad (7)$$

From the above expression, we obtain b_{max} as

$$b_{max} = \frac{\{ 2 * (n-2) * 3^a - (n-1) * 2^a \}}{\left\{ 2^a * \sum_{j=r+1}^n (j-1)(n-j+1) - 3^a * \sum_{j=r+1}^n (j-1)(n-j+1) \right\}} \quad (8)$$

This above b_{max} is the bound b^* . We can easily see that if $b > b^*$ then the optimal sequence is obtained by arranging the longest job in first position and the rest of the jobs in SPT order.

From the b_{max} expression, we can also find the bound on learning index a . We know that $b \geq 0$. We find the value of a for which $b_{max} = 0$ and this value of a is the bound on learning index a^* . This is obtained as

$$2 * (n-2) * 3^a = (n-1) * 2^a \quad (9)$$

From which we obtain

$$\begin{aligned} \frac{2^a}{3^a} &= \frac{2 * (n-2)}{(n-1)} \\ a \{ \log(2) - \log(3) \} &= \{ \log(2 * (n-2)) - \log(n-1) \} \end{aligned} \quad (10)$$

Hence, we obtain

$$a^* = \frac{\{ \log(2 * (n-2)) - \log(n-1) \}}{\{ \log(2) - \log(3) \}} \quad (11)$$

Here also, we can see that if $a < a^*$ then the optimal sequence is obtained by arranging the longest job in first position and the rest of the jobs in SPT order.

Effect of learning index a : The number of optimal sequences and the range depends on the value of a in addition to the value of b . For the numerical example $n = 7$, if the value of $a = -0.8$, we obtain only three sequences that are optimal. Our computational algorithm will find the optimal sequences and the range of b in which each of these sequences are optimal. The results are shown in Table.2. The reason for this is that some of the lines v_r will intersect for values of $b < 0$, which is not a feasible solution.

Table 2: Range of b and the optimal sequence for $1/LE, s_{psd}/TADC$ problem ($a = -0.80$)

Range of b	Optimal Sequence
$0 < b < 0.0263376$	{7, 3, 1, 2, 4, 5, 6}
$0.0263376 < b < 0.053298$	{7, 2, 1, 3, 4, 5, 6}
$b > 0.0583298$	{7, 1, 2, 3, 4, 5, 6}

4. Conclusions

We considered the single machine scheduling problems with past-sequence-dependent setup times and a learning effect. The setup times are proportional to the length of jobs that are already scheduled; i.e., past-sequence-dependent (p-s-d) setup times. The actual processing time of a job depends on its position in the sequence because of the learning effect. In this paper, we presented a parametric analysis of b on the $1/LE, s_{psd}/TADC$ problem for a given value of a . A computational algorithm is presented to obtain the number of optimal sequences and the range of b in which each of the sequences is optimal, for a given value of a . Two bounds b^* for the

normalizing constant b and a^* for the learning index a are derived. It is shown that when $a < a^*$ or $b > b^*$, the optimal sequence is obtained by arranging the longest job in first position and the rest of the jobs in SPT order.

References

- [1] B. Alidaee, F. Landram, Scheduling deteriorating jobs on a single machine to minimize the maximum processing times, *International Journal of Systems Science*, 27 (1996) 507–510.
- [2] U. Bagchi, Simultaneous minimization of mean and variance of flow-time and waiting time in single machine systems, *Oper.Res.* 37 (1989) 118-125.
- [3] D. Biskup, Single-machine scheduling with learning considerations, *Eur.J.Oper.Res.* 115 (1999) 173-178.
- [4] D. Biskup, A state-of-the-art review on scheduling with learning effects, *Eur.J.Oper.Res.* 188 (2008) 315-329.
- [5] T.C.E. Cheng, G. Wang, Single machine scheduling with learning effect considerations, *Ann.Oper.Res.* 98 (2000) 273-290.
- [6] T.C.E. Cheng, Q. Ding, B.M.T. Lin, A concise survey of scheduling with time-dependent processing times, *Eur.J.Oper.Res.* 152 (2004) 1-13.
- [7] G.H. Hardy, J.E. Littlewood, G. Polya, *Inequalities*. Cambridge University Press, London, 1967.
- [8] C. Koulamas, G.J. Kyparisis, Single machine scheduling with past-sequence-dependent setup times, *Eur.J.Oper.Res.* 187 (2008) 1045-1049.

- [9] W.H. Kuo, D.L. Yang, Single machine scheduling with past-sequence-dependent setup times and learning effects, *Info. Processing Letters*, 102 (2007) 22-26.
- [10] W.C. Lee, C.C. Wu, Minimizing total completion time in a two-machine flow-shop with a learning effect, *Intl.J.Prod.Economics*, 88 (2004) 85-93.
- [11] G. Mosheiov, Scheduling problems with a learning effect, *Eur.J.Oper.Res.* 132 (2001) 687-693.
- [12] G. Mosheiov, Parallel machine scheduling with a learning effect, *J.Oper.Res.Soc.* 52 (2001) 1165-1169.
- [13] R. Uzsoy, C. Lee, and L. A. Martin-Vega, Scheduling semiconductor test operations, minimizing maximum lateness and number of tardy jobs on a single machine, *Naval Research Logistics*, 39 (1992) 369-388.
- [14] J. Wang, A note on scheduling problems with learning effect and deteriorating jobs, *International Journal of Systems Science* 37 (2006), 827–833.