International Journal of Systems Science
Vol. 42, No. 12, December 2011, 2097–2102

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

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

[a]*Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India;* [b]*Department of Information Management, Yuan Ze University, 135, Yuan-Dong Road, Tao-Yuan 320, Taiwan, R.O.C.;* [c]*Department of Electronic Commerce Management, Nanhua University, Chiayi 62248, Taiwan, R.O.C.*

(*Received 23 January 2009; final version received 17 February 2010*)

In this article, we consider the single-machine scheduling problem with past-sequence-dependent (p-s-d) setup times and a learning effect. The setup times are proportional to the length of jobs that are already scheduled; i.e. 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 Kuo and Yang (2007) ('Single Machine Scheduling with Past-sequence-dependent Setup Times and Learning Effects', *Information Processing Letters*, 102, 22–26). There are two parameters a and b denoting constant learning index and normalising index, respectively. A parametric analysis of b on the $1/LE, s_{psd}/TADC$ problem for a given value of a is applied in this study. In addition, a computational algorithm is also 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 normalising 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 short processing time order.

Keywords: scheduling; setup times; learning effect

1. Introduction

As mentioned in Alidaee and Landram (1996), scheduling problems in many real-world applications have the characteristic of variable processing time, that is the processing time of a job is a variable and depends on a function of its starting time. In a recent study, Koulamas and Kyparisis (2008) introduce the concept of past-sequence-dependent (p-s-d) setup times in single-machine scheduling problems. In fact, the study is the first to consider the p-s-d 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 workers to learn and improve their performance. So the processing time of a job reduces due to learning. Biskup (1999) was the first to address the effect of learning in the context of single-machine scheduling problems. It is shown by Biskup (1999) that this problem can be solved in polynomial time if the objectives are minimisation 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 minimising the flow time are considered in Mosheiov (2001a,b). The learning effect in a two machine flowshop scheduling with the objective of finding the sequence of jobs that minimises the total completion time (TC) is given by Lee and Wu (2004). In Lee and Wu (2004), a branch and bound technique is presented. A heuristic algorithm is also presented in Lee and Wu (2004) to improve the efficiency of the branch and bound technique. Cheng and Wang (2000) consider the learning effect on the processing time of jobs using a volume dependent processing time function model. Wang (2006) mentions in his recent study that some single-machine scheduling problems remain polynomially solvable when deterioration and learning effect on job processing times are involved. Cheng, Ding, and Lin (2004) present a concise survey of scheduling with time-dependent processing times. In a recent study, Biskup (2008) presents a complete discussion on why and when the learning effects might occur and a concise review of the literature on scheduling with learning effects. Cheng, Wang, and He (2009) further consider the scheduling problems on parallel identical machines with deteriorating jobs, in which the processing time of a job is a proportional function of its

*Corresponding author. Email: iepchang@saturn.yzu.edu.tw

ISSN 0020–7721 print/ISSN 1464–5319 online

http://dx.doi.org/10.1080/00207721003718436
http://www.tandfonline.com

starting times to be processed. They construct heuristic algorithms for this parallel identical machine scheduling problem and also analyse the performance of these algorithms.

In this article, we consider the non-preemptive single-machine scheduling problems with p-s-d setup times along with learning effect. To the best of our knowledge, Kuo and Yang (2007) were the first to study the concept of p-s-d setup times along with learning effect in single-machine scheduling problems. The objectives considered by Kuo and Yang (2007) are minimising the maximum completion time ($C_{\max}$), TC, total absolute difference in completion times (TADC), and the unit earliness, tardiness and due date penalty (ETCP). These scheduling problems with p-s-d setup times along with learning effect are denoted in Kuo and Yang (2007) 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 is 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, $(r = 1, 2, \ldots, n)$. The processing time of a job after learning and occupying position r in the sequence is given by

$$p^l_{[r]} = p_{[r]} r^a, \quad n = 1, 2, \ldots, n, \tag{1}$$

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

$$\begin{aligned} s_{[1]} &= 0, \\ s_{[r]} &= b \sum_{j=1}^{r-1} p^l_{[j]} \quad r = 2, 3, \ldots, n, \end{aligned} \tag{2}$$

where $b \geq 0$ is a normalising constant. In Equation (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 Kuo and Yang (2007) that the well-known shortest processing time (SPT) sequence is optimal for both Problems (i) and (ii).

1.1. *Contributions of this article*

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 normalising 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.

In terms of the contribution for the industry, Koulamas and Kyparisis (2008) indicate that the consideration of p-s-d 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, Lee, and Martin-Vega (1992) mentioned a more general manufacturing environment in which 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 minimising the TADC. The $TADC$ of the $1/LE,\ s_{psd}/TADC$ scheduling problem given in Kuo and Yang (2007) is

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

As mentioned in Kuo and Yang (2007), the Equation (3) can be viewed as the scalar product of two vectors. One vector is $p_{[r]}$, that is the vector of the processing time of jobs. The other is v_r, which is known as the positional weights vector and is given as

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

In Equation (4), the value of $v_1 = 0$ because $s_{[1]}$ is zero ($\because s_{[r]} = b \sum_{j=1}^{r-1} P^A_{[j]}$ and v_1 is an initial weight, see also Kuo and Yang (2007)). It is well known from Hardy, Littlewood, and Polya (1967) that Equation (3) is minimised by arranging the vectors v_r and $p_{[r]}$ in opposite orders. This is also given in Kuo and Yang (2007) 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.

2.1. *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 Equation (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 article.

2.2. *Motivating numerical example*

Let us consider the 7-job example given in Bagchi (1989). 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 Bagchi (1989). For this numerical example the positional weights are:

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

In order to study the effect of b on the optimal sequence, we plot the above values of positional weights v_r, $(r = 1, 2, \ldots, n)$, with the value of b. The variations of v_r, $(r = 1, 2, \ldots, n)$ for b values in the range of (0, 0.5) are 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 $v_1, v_2, \ldots, v_7$. 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 by using Hardy et al. (1967) is

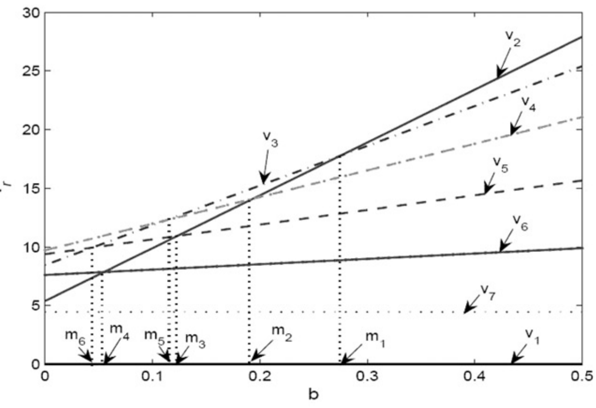

Figure 1. Variation of v_r as a function of b.

$\{7, 2, 1, 3, 4, 5, 6\}$. 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 by using Hardy et al. (1967) is $\{7, 2, 1, 3, 4, 5, 6\}$. This implies that in this range of b (0.2–0.25) the optimal sequence is unique. Also, note that in this range of b (0.2–0.25) all the values of v_r for $r = 2, 3, \ldots, 6$ increase with 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, $(r = 1, 2, \ldots, n)$ given by Equation (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 $5.4000 + 45.0000 \times b = 8.4620 + 33.8484 \times b$. From this we get $11.152 \times b = 3.062$ and hence $b = 0.2746$. There are six points of intersection for this example ($n = 7$), denoted as m_1–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 $m_5 = 0.1126$ and lines v_3 and v_5 will intersect at point $m_6 = 0.0438$.

We arrange these six intersection points m_1–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 Hardy et al. (1967) $\{7, 2, 1, 3, 4, 5, 6\}$. This sequence is optimal in the range of b given by m_4 and m_5. In this manner, we obtain seven optimal sequences. The optimal sequences and the range of b

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

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

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.

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 is 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 the 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, $(r = 1, 2, \ldots, n)$ for $b > 0$. The intersection points give the range of b. Once the intersection points are obtained, the optimal sequence is proposed by Hardy et al. (1967). 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**
  $A(r) = (r-1)(n-r+1) \times r^a$
  $B(r) = \{\sum_{j=r+1}^{n} (j-1)(n-j+1)\} \times r^a$
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 (Hardy et al. 1967). Choose one value of b in the range $0 < b < b_{\min}$ and obtain the optimal sequence using Hardy et al. (1967). Also choose one value of b in the range $b > b_{\max}$ and obtain the optimal sequence in same manner using Hardy et al. (1967).

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.

3.1. *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

$$v_2 = \left\{(n-1) + b \times \sum_{j=r+1}^{n} (j-1)(n-j+1)\right\} \times 2^a,$$
$$v_3 = \left\{2 \times (n-2) + b \times \sum_{j=r+1}^{n} (j-1)(n-j+1)\right\} \times 3^a. \tag{6}$$

The intersection point of lines v_2 and v_3 is

$$\left\{(n-1) + b \times \sum_{j=r+1}^{n} (j-1)(n-j+1)\right\} \times 2^a = \left\{2 \times (n-2) + b \times \sum_{j=r+1}^{n} (j-1)(n-j+1)\right\} \times 3^a.$$

This reduces to

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

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

$$b_{\max} = \frac{\left\{2\times(n-2)\times 3^a - (n-1)\times 2^a\right\}}{\left\{\begin{array}{l} 2^a \times \sum_{j=r+1}^{n}(j-1)(n-j+1) \\ \quad -3^a \times \sum_{j=r+1}^{n}(j-1)(n-j+1)\end{array}\right\}}. \quad (8)$$

This $b_{\max}$ 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

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

From which we obtain

$$\frac{2^a}{3^a} = \frac{2\times(n-2)}{(n-1)},$$

$$a\left\{\log(2) - \log(3)\right\} = \left\{\log(2\times(n-2)) - \log(n-1)\right\}. \quad (10)$$

Hence, we obtain

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

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.

3.2. *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 p-s-d setup times and a learning effect. The setup times are proportional to the length of jobs that are already scheduled; i.e. 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 article, a parametric analysis of b on the $1/LE$, $s_{psd}/TADC$ problem for a given value of a is presented. 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 normalising 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 the SPT order.

Notes on contributors

V. Mani received his BE degree in Civil Engineering from Madurai University in 1974, his MTech degree in Aeronautical Engineering from the Indian Institute of Technology, Madras in 1976 and his PhD degree in Engineering from the Indian Institute of Science, Bangalore in 1986. From 1986 to 1988, he was a Research Associate at the School of Computer Science, University of Windsor, Windsor, ON, Canada, and from 1989 to 1990 at the Department of Aerospace Engineering, Indian Institute of Science, Bangalore. Since 1990, he has been with the Indian Institute of Science, Bangalore, where he is presently a Professor in the Department of Aerospace Engineering. His research interests include scheduling, evolutionary computation, parallel and distributed computing, queueing networks and mathematical modelling. He is one of the authors of a book, Scheduling Divisible loads in Parallel and Distributed Systems (Los Alamitos, CA: IEEE Computer Society Press).

Pei-Chann Chang received his MS and PhD degrees from the Department of Industrial Engineering of Lehigh University in 1985 and 1989, respectively. He is a Professor in Yuan-Ze University in Taiwan. His research interests include production scheduling, sales forecasting, case-based reasoning, ERP, global logistics and applications of soft computing. He has published his research works in international journals, such as *Decision Support Systems*, *Expert Systems with Applications*, *European Journal of Operational Research*, *International Journal of Production Economics*, *Applied Soft Computing*, *Journal of Intelligent Manufacturing* and *Computers and Operations Research*.

Shih-Hsin Chen received his PhD degree from the Department of Industrial Engineering of Yuan-Ze University in Taiwan. He is currently an Assistant Professor in the Department of Electronic Commerce Management, Nanhua University, Taiwan. His research interests are in multi-objective problems in production scheduling, soft computing applied in manufacturing problems and development of genetic algorithms. He has published his research works in some international journals, such as *Expert Systems with Applications*, *Applied Soft Computing*, *Applied Mathematical Modeling*, *Annals of Operations Research* and *Applied Mathematics and Computation*.

References

Alidaee, B., and Landram, F. (1996), 'Scheduling Deteriorating Jobs on a Single Machine to Minimize the Maximum Processing Times', *International Journal of Systems Science*, 27, 507–510.

Bagchi, U. (1989), 'Simultaneous Minimization of Mean and Variance of Flow-time and Waiting Time in Single Machine Systems', *Operation Research*, 37, 118–125.

Biskup, D. (1999), 'Single-machine Scheduling with Learning Considerations', *European Journal of Operation Research*, 115, 173–178.

Biskup, D. (2008), 'A State-of-the-art Review on Scheduling with Learning Effects', *European Journal of Operation Research*, 188, 315–329.

Cheng, T.C.E., Ding, Q., and Lin, B.M.T. (2004), 'A Concise Survey of Scheduling with Time-dependent Processing Times', *European Journal of Operation Research*, 152, 1–13.

Cheng, M., Wang, G., and He, L. (2009), 'Parallel Machine Scheduling Problems with Proportionally Deteriorating Jobs', *International Journal of Systems Science*, 40, 53–57.

Cheng, T.C.E., and Wang, G. (2000), 'Single Machine Scheduling with Learning Effect Considerations', *Annals of Operation Research*, 98, 273–290.

Hardy, G.H., Littlewood, J.E., and Polya, G. (1967), *Inequalities*, London: Cambridge University Press.

Koulamas, C., and Kyparisis, G.J. (2008), 'Single Machine Scheduling with Past-sequence-dependent Setup Times', *European Journal of Operation Research*, 187, 1045–1049.

Kuo, W.H., and Yang, D.L. (2007), 'Single Machine Scheduling with Past-sequence-dependent Setup Times and Learning Effects', *Information Processing Letters*, 102, 22–26.

Lee, W.C., and Wu, C.C. (2004), 'Minimizing Total Completion Time in a Two-machine Flowshop with a Learning Effect', *International Journal of Economics*, 88, 85–93.

Mosheiov, G. (2001a), 'Scheduling Problems with a Learning Effect', *European Journal of Operation Research*, 132, 687–693.

Mosheiov, G. (2001b), 'Parallel Machine Scheduling with a Learning Effect', *Journal of the Operational Research Society*, 52, 1165–1169.

Uzsoy, R., Lee, C., and Martin-Vega, L.A. (1992), 'Scheduling Semiconductor Test Operations, Minimizing Maximum Lateness and Number of Tardy Jobs on a Single Machine', *Naval Research Logistics*, 39, 369–388.

Wang, J. (2006), 'A Note on Scheduling Problems with Learning Effect and Deteriorating Jobs', *International Journal of Systems Science*, 37, 827–833.