ELSEVIER
European Journal of Operational Research 132 (2001) 687–693
www.elsevier.com/locate/dsw

Theory and Methodology

# Scheduling problems with a learning effect

Gur Mosheiov [*]

*School of Business Administration and Department of Statistics, The Hebrew University of Jerusalem, Mount Scopus, Jerusalem 91905, Israel*

Received 12 November 1999; accepted 25 May 2000

**Abstract**

In many realistic settings, the production facility (a machine, a worker) improves continuously as a result of repeating the same or similar activities; hence, the later a given product is scheduled in the sequence, the shorter its production time. This "learning effect" is investigated in the context of various scheduling problems. It is shown in several examples that although the optimal schedule may be very different from that of the classical version of the problem, and the computational effort becomes significantly greater, polynomial-time solutions still exist. In particular, we introduce polynomial solutions for the single-machine makespan minimization problem, and two for multi-criteria single-machine problems and the minimum flow-time problem on parallel identical machines. © 2001 Elsevier Science B.V. All rights reserved.

*Keywords:* Scheduling; Single-machine; Parallel machines; Learning

## 1. Introduction

Consider a sequential production of different products by a standard production facility. A common assumption in traditional scheduling is that the production time of a given product is independent of its position in the production sequence. However, in many realistic settings, the production facility (a machine, a worker) improves continuously with time. As a result, the production time of a given product is shorter if it is scheduled later, rather than earlier in the sequence. This phenomenon is known in the literature as a "learning effect".

Different types of learning effects have been demonstrated and extensively studied in a number of areas. However, to the best of our knowledge, apart from the recent paper of Biskup (1999), it has not been investigated in the context of production scheduling. Biskup considered a single-machine setting and assumed two objectives: (i) minimizing flow-time, and (ii) minimizing the weighted sum of completion-time deviations from a common due-date and the sum of job completion-times. The learning curve assumed in Biskup's paper reflects decrease in the production time as a function of the number of repetitions (see Nadler

[*] Corresponding author. Tel.: +972-2-588-3790; fax: +972-2-588-1341.
*E-mail address:* msomer@mscc.huji.ac.il (G. Mosheiov).

and Smith, 1963; Yelle, 1979). Biskup introduced polynomial-time solutions for both scheduling problems.

The present study investigates several scheduling problems with learning considerations, using the learning curve introduced by Biskup (1999). First, we focus on a comparison of the optimal schedule of some classical single-machine problems, with and without a learning effect. Biskup showed that the minimum flow-time problem is solved by the smallest processing time first (SPT) schedule, in both cases. We show that SPT also solves the makespan minimization problem with a learning effect (unlike the classical version in which the makespan value is sequence-independent). Three other objectives, minimizing the sum of weighted completion times, minimizing maximum lateness and minimizing the number of tardy jobs, appear to be more complicated. We show that the $O(n \log n)$ (where $n$ is the number of jobs) solutions of the classical version do not hold with learning considerations.

We then solve two single-machine multi-criteria problems (i.e., we seek a schedule that performs well with respect to several measures). First, we consider the classical due-date assignment problem introduced by Panwalker et al. (1982). The objective is of minimizing a weighted sum of earliness, tardiness and due-date costs. Then, we study a problem in which the objective is a linear combination of the sum of job completion times and the sum of job completion-time deviations (Bagchi, 1989). The original versions of the two problems (i.e., with no learning effect) have polynomial-time ($O(n \log n)$) solutions. We show that with a learning effect, both problems remain polynomially solvable, but significantly larger computational effort is required. (Both problems are presented as assignment problems, and thus an $O(n^3)$ effort is required for the solution.) Finally, we discuss learning effect in settings of parallel identical machines. While makespan minimization remains NP-hard (as in the classical version), it is not clear whether flow-time minimization remains easy. We show that the SPT policy (which is optimal in the classical version of flow-time minimization) is not optimal when learning effect is assumed.

In Section 2, we focus on the classical single-machine problems. In Section 3, we provide polynomial solutions for the multi-criteria problems. Section 4 is devoted to the minimum flow-time problem on parallel machines.

## 2. Classical objectives

$n$ jobs are to be processed on a single machine. In classical scheduling, $p_j$ denotes the (sequence-independent) processing time of job $j$ ($j = 1, \ldots, n$). Assume that jobs are numbered such that $p_1 \leqslant p_2 \leqslant \cdots \leqslant p_n$ (an SPT sequence). For a given schedule $q$, $C_j = C_j(q)$ represents the completion time of job $j$.

Assume now that the production facility improves continuously, and that the processing time of a given job decreases as a function of its position in the sequence. As in Biskup (1999), we assume that the processing time of job $j$ if scheduled in position $r$, is given by

$$p_{jr} = p_j r^a, \quad j, r = 1, \ldots, n, \tag{1}$$

where $a \leqslant 0$ is a constant learning index. For convenience, we denote the learning effect given in (1) by $LE$. Thus, using the conventional notation for describing scheduling problems, we denote, e.g., the single-machine makespan minimization problem by $1/LE/C_{\max}$. In the following, we examine several classical single-machine scheduling problems with the above learning effect. The first two problems have polynomial-time solutions.

### 2.1. Minimum makespan

A standard pair-wise interchange argument proves that $1/LE/C_{\max}$ is optimized by the SPT sequence. Recall that in the classical makespan minimization problem ($1//C_{\max}$), the makespan value is sequence-independent.

### 2.2. Minimum flow-time

Biskup (1999) used a pair-wise interchange argument to prove that $1/LE/\sum C_j$ is optimized by an SPT sequence. This result coincides with the optimal policy for $1//\sum C_j$.

In the next three problems, we show that the optimal schedule of the classical version is not optimal when a learning effect is assumed.

### 2.3. Minimum sum of weighted completion times

$1//\sum W_jC_j$ (where $W_j$, $j = 1, \dots, n$, are job weights) is known to be optimally solved by Smith's rule (1956); also known as weighted smallest processing time first or WSPT), that is, by a job sequence of increasing $p_j/W_j$-values. In the following example, we show that this policy is not optimal for $1/LE/\sum W_jC_j$:

**Example 1.** □ = 2, □₁ = 1, □₂ = 2, □₁ = 10, □₂ = 21. Assume, as in Biskup (1999), an 80%-learning curve, i.e., □ = (learning index = )−0.322. The cost of the sequence obtained by WSPT (job 2 followed by job 1) is 70. The cost of the alternative (optimal) sequence is 64.6.

### 2.4. Minimum maximum lateness

If $d_j$ denotes the due-date of job $j$, and $L_j = C_j - d_j$ is its lateness, $j = 1, \dots, n$, then problem $1//L_{\max}$ is known to be solved by the earliest due-date first (EDD) policy. The example below shows that this policy is not optimal for $1/LE/L_{\max}$.

**Example 2.** □ = 2, □₁ = 1, □₂ = 100, □₁ = 1, □₂ = 0, □ = −0.322. □max obtained by the EDD sequence (job 2 followed by job 1) is 100. The optimal □max value obtained by the sequence (1,2) is 81.

### 2.5. Minimum number of tardy jobs

Let $U_j = 1$ if $C_j > d_j$ (i.e., the job is late) and $U_j = 0$ otherwise, $j = 1, \dots, n$. The problem $1//\sum U_j$ is known to be solved by Moore's Algorithm (1968). As a special case, it is known (Jackson, 1955) that if a schedule with no tardy jobs exists, then the EDD sequence will contain no tardy jobs. Example 3 shows that Jackson's lemma does not hold for $1/LE/\sum U_j$ (therefore, Moore's Algorithm is not optimal for the problem).

**Example 3.** □ = 2, □₁ = 1, □₂ = 100, □₁ = 91, □₂ = 90, □ = −0.322. The sequence (1,2) contains no tardy jobs. The EDD sequence (2,1) contains two tardy jobs.

As expected, in many problem types, the learning effect has a major impact on the optimal policy. In general (but not always, see e.g., $1/LE/\sum C_j$), the computational effort required for solving scheduling problems with a learning effect increases (see also Section 3). However, we note that the question of the complexity status of the last three problems remains open. Possible heuristics for these problems may be based on the optimal schedules for the classical versions. Thus, an initial solution for $1/LE/\sum W_jC_j$ could be WSPT, an initial solution for $1/LE/L_{\max}$ could be EDD, and an initial solution for $1/LE/\sum U_j$ could be obtained by Moore's Algorithm. Then, an improvement procedure based on e.g., job replacement may be used.

## 3. Multi-criteria problems

In this section, we study two single-machine multi-criteria problems, with learning effects. Although the computational effort is significantly greater when a learning effect is assumed, in both cases, the optimal solution procedure remains polynomial.

### 3.1. A due-date assignment problem (Panwalker et al., 1982)

Consider an $n$-job single-machine problem, with $p_j$ denoting the (sequence-independent) processing time of job $j$ $(j = 1, \dots, n)$. All jobs share a common due-date $d$, yet to be determined. For a given schedule $q, C_j = C_j(q)$ represents the completion time of job $j$, $E_j = \max\{0, d - C_j\}$ is the earliness value of job $j$, and $T_j = \max\{0, C_j - d\}$ is the tardiness value of job $j$, $j = 1, \dots, n$. Let $\alpha$, $\beta$ and $\gamma$

denote the unit penalty for due-date delay, earliness and tardiness, respectively. The objective function of both the due-date $d$ and the schedule $q$ is

$$f(d,q) = \sum_{j=1}^{n} (\alpha d + \beta E_j + \gamma T_j). \tag{2}$$

Panwalker et al. showed that (i) for any given schedule, it is optimal to assign the due-date at the completion time of the $k$th job, where

$$k = \left\lceil \frac{\gamma - \alpha}{\beta + \gamma} \right\rceil \tag{3}$$

and (ii) the optimal schedule is V-shaped, i.e., early jobs are arranged in a non-increasing order of processing times, and tardy jobs are arranged in a non-decreasing order of processing times. The positional weight of position $r$ in the sequence is given by

$$w_r = \begin{cases} n\alpha + (r-1)\beta & \text{if } r \leqslant k, \\ (n+1-r)\gamma & \text{if } r > k. \end{cases} \tag{4}$$

Thus, the optimal schedule 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 smaller positional weight, etc.

Now assume a learning effect as given in (1). In this new setting, our objective remains to find $d$ and $q$ that minimize (2). Using a similar model to that of Biskup (1999), let $x_{jr}$ be a 0/1 variable such that $x_{jr} = 1$ if job $j$ is scheduled in position $r$, and $x_{jr} = 0$, otherwise. As in Biskup, the optimal matching of jobs to positions requires a solution for the following assignment problem:

$$\min \quad \sum_{j=1}^{n}\sum_{r=1}^{n} w_r p_{jr} x_{jr} \tag{5}$$

subject to

$$\sum_{j=1}^{n} x_{jr} = 1, \quad r = 1,\ldots,n,$$

$$\sum_{r=1}^{n} x_{jr} = 1, \quad j = 1,\ldots,n,$$

$$x_{jr} = 0 \text{ or } 1, \quad j,r = 1,\ldots,n.$$

Recall that solving an assignment problem of size $n$ requires an effort of $O(n^3)$. Once the assignment problem is solved, we obtain an optimal matching of jobs to positions (i.e., an optimal sequence). Note that the value of $k$ (see (3)) is independent of the actual processing times of the jobs and, hence, it is not affected by the learning effect (see Biskup, 1999). Thus, the optimal due-date is at the completion time of the $k$th job in the sequence obtained from the solution of the assignment problem. In order to demonstrate the above, we solve the instance introduced (and solved for the case of no learning effect) by Panwalker et al. with a learning effect assumption:

**Example 4** (Panwalker et al.). Data: $n=7$, $p_1=3$, $p_2=4$, $p_3=6$, $p_4=9$, $p_5=14$, $p_6=18$, $p_7=20$, $\alpha=5$, $\beta=11$, $\gamma=18$. The optimal $k$-value is 4. The positional weights are: $w_1=35$, $w_2=46$, $w_3=57$, $w_4=68$, $w_5=54$, $w_6=36$, $w_7=18$. The optimal sequence (with no learning effect) is (6,4,2,1,3,5,7). Thus, the optimal due-date is at time $p_6 + p_4 + p_2 + p_1 = 34$, and the total cost is 2664. Assume now an 80%-learning curve, i.e., $a = -0.322$. The input for the assignment problem (5), i.e., job/location processing times and positional weights, are given in Table 1. The solution of the assignment problem (obtained by the Solver of EXCEL) leads to a new optimal sequence: (4,3,2,1,5,6,7). The optimal $k$-value remains 4. The optimal due-date is at

$$p_4 1^{-0.322} + p_3 2^{-0.322} + p_2 3^{-0.322} + p_1 4^{-0.322} = 18.53,$$

and the total cost is 1832.97. The optimal sequence is clearly different from the optimal sequence in the original version of the problem. Note that both are V-shaped with respect to the $p_i$-values. The processing time decreased due to the learning effect, and, therefore, as expected, both the optimal due-date value and the total cost are smaller.

### *3.2. Simultaneous minimization of total completion time and variation of completion times (Bagchi, 1989)*

In this single-machine bi-criteria problem, we look for a schedule that performs well with respect

Table 1
Job/location processing times and positional weights for the due-date assignment problem of Panwalker et al. (1982)[a]

| $j$ | $r$ | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 3.0 | 2.4 | 2.1 | **1.9** | 1.8 | 1.7 | 1.6 |
| 2 | 4.0 | 3.2 | **2.8** | 2.6 | 2.4 | 2.2 | 2.1 |
| 3 | 6.0 | **4.8** | 4.2 | 3.8 | 3.6 | 3.4 | 3.2 |
| 4 | **9.0** | 7.2 | 6.3 | 5.8 | 5.4 | 5.1 | 4.8 |
| 5 | 14.0 | 11.2 | 9.8 | 9.0 | **8.3** | 7.9 | 7.5 |
| 6 | 18.0 | 14.4 | 12.6 | 11.5 | 10.7 | **10.1** | 9.6 |
| 7 | 20.0 | 16.0 | 14.0 | 12.8 | 11.9 | 11.2 | **10.7** |
| $w_r$ | 35 | 46 | 57 | 68 | 54 | 36 | 18 |

[a] The optimal sequence ((4,3,2,1,5,6,7); see bold numbers) is obtained by solving the associated assignment problem. $j$ – job index; $r$ – location index, $w_r$ – positional weight.

to both a classical efficiency measure (total completion time) and a measure of performance balance (variation of completion times). As in Section 2, $p_j$ denotes the (sequence-independent) processing time of job $j$, and (for a given schedule) $C_j$ represents the completion time of job $j$, $j = 1, \ldots, n$. TC denotes the total completion time $\text{TC} = \sum_{j=1}^{n} C_j$. TADC denotes the total absolute differences in completion times $\text{TADC} = \sum_{i=1}^{n} \sum_{j=1}^{n} |C_i - C_j|$. Let $0 \leqslant \delta \leqslant 1$. The objective is to find a schedule that minimizes the linear combination of both measures:

$$f(q) = \delta\text{TC} + (1 - \delta)\text{TADC}. \tag{6}$$

Bagchi showed that the positional weight of position $r$ in the sequence is given by

$$\begin{aligned} w_r = {} & (2\delta - 1)(n + 1) + r[2 - 3\delta + n(1 - \delta)] \\ & - r^2(1 - \delta), \\ r = {} & 1, \ldots, n. \end{aligned} \tag{7}$$

The optimal schedule is obtained, as in Section 2, by the same matching procedure of jobs to positions.

When the learning effect specified in (1) is assumed, we have to solve the assignment problem (5) (with the appropriate positional weights). In the following, we solve (with learning effect assumptions) the instance introduced by Bagchi:

**Example 5** □□□□□□□). Data: □ = 7, $□_1 = 2$, $□_2 = 3$, $□_3 = 6$, $□_4 = 9$, $□_5 = 21$, $□_6 = 65$, $□_7 = 82$, $\delta = 0.5$. The positional weights are $□_1 = 3.5$, $□_2 = 6$, $□_3 = 7.5$, $□_4 = 8$, $□_5 = 7.5$, $□_6 = 6$, $□_7 = 3.5$. An optimal sequence (with no learning effect) is (7,5,3,1,2,4,6), with cost 1139.

With a learning effect (an 80%-learning curve is assumed, i.e., $a = -0.322$), the problem can be formulated as an assignment problem. The input (location-dependent processing times and positional weights) is given in Table 2. We obtain a new optimal sequence: (5,3,1,2,4,6,7). The total cost is 540.79. Again, the optimal sequence is different from the original optimal sequence (although both are V-shaped with respect to the $p_j$-values), and the total cost is significantly smaller.

## 4. Scheduling with learning effect on Parallel Identical Machines

With no learning effect, $Pm//C_{\max}$ (where $m$ is the number of parallel identical machines) is known to be NP-hard even for two machines. Clearly, $Pm/LE/C_{\max}$ is also NP-hard since the special case $a = 0$ is identical with the classical version. Minimizing flow-time on parallel identical machines ($Pm//\Sigma C_j$) is solved by the SPT policy. Similar to the single-machine case, when learning effect is assumed, one can easily show that an optimal schedule exists with jobs sequenced in a non-decreasing order of processing times on each machine:

**Proposition 1.** An optimal schedule for $Pm/LE/\Sigma C_j$ consists of SPT sequences on each machine.

Table 2
Job/location processing times and positional weights for the bi-criteria problem of Bagchi (1989)[a]

| $j$ | $r$ | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 2.0 | 1.6 | **1.4** | 1.3 | 1.2 | 1.1 | 1.1 |
| 2 | 3.0 | 2.4 | 2.1 | **1.9** | 1.8 | 1.7 | 1.6 |
| 3 | 6.0 | **4.8** | 4.2 | 3.8 | 3.6 | 3.4 | 3.2 |
| 4 | 9.0 | 7.2 | 6.3 | 5.8 | **5.4** | 5.1 | 4.8 |
| 5 | **21.0** | 16.8 | 14.7 | 13.4 | 12.5 | 11.8 | 11.2 |
| 6 | 65.0 | 52.0 | 45.6 | 41.6 | 38.7 | **36.5** | 34.7 |
| 7 | 82.0 | 65.6 | 57.6 | 52.5 | 48.8 | 46.1 | **43.8** |
| $w_r$ | 3.5 | 6 | 7.5 | 8 | 7.5 | 6 | 3.5 |

[a] The optimal sequence ((5,3,1,2,4,6,7); see bold numbers) is obtained by solving the associated assignment problem. $j$ – job index; $r$ – location index, $w_r$ – positional weight.

**Proof**. We slightly modify the proof of Biskup (1999) for the single machine case. Let $q$ denote a sequence (on any machine) with job $j$ in position $r$ and job $k$ in position $r+1$, and let $s$ denote the same sequence with jobs $j$ and $k$ in opposite positions. Let $C_j(q)$ denote the completion time of job $j$ in sequence $q$, and let $F(q)$ denote the total flow time of sequence $q$. Denote by $t$ the starting time of job $j(k)$ in sequence $q(s)$. Assume $p_j \leqslant p_k$. In order to prove optimality of SPT, i.e. $F(q) \leqslant F(s)$, we have to show that (1) $C_j(q) + C_k(q) \leqslant C_k(s) + C_j(s)$, and (2) $C_k(q) \leqslant C_j(s)$. (1) guarantees that the contribution to the total flow-time of jobs $j$ and $k$ in sequence $q$ is less than or equal to their contribution in sequence $s$. (2) guarantees that all the jobs scheduled in $q$ after the pair of jobs $j$ and $k$, have completion times not larger than their completion times in $s$. (1) is verified as in Biskup by a pairwise interchange argument $(t+p_jr^a+t+p_jr^a+p_k(r+1)^a \leqslant t+p_k r^a+t+p_kr^a+p_j(r+1)^a)$. (2) is true since SPT minimizes makespan on a single machine, see Section 2. (The appropriate pairwise interchange implies $(t+p_jr^a+ p_k(r+1)^a \leqslant t+p_kr^a+p_j(r+1)^a)$. □

Despite this property, the following example shows that an optimal schedule for $Pm/LE/\Sigma C_j$ is *not* necessarily SPT. (Recall that an SPT schedule on parallel identical machines consists of scheduling the jobs according to an SPT list, where the following job on the list is scheduled on the next available machine.)

**Example 6.** Data: $m=2, n=3, p_1=1,\ p_2=2,\ p_3=100, a=-0.322$. Total flow-time obtained by the SPT schedule (jobs 1 and 3 on machine 1, and job 2 on machine 2) is $1+[1+100\ (2^{-0.322})]+2=84$. An optimal schedule is obtained by scheduling jobs 1, 2 and 3 on machine 1 and no jobs on machine 2. The optimal flow-time value is $1+[1+2(2^{-0.322})+[1+2(2^{-0.322})]+100(3^{-0.322})] = 76.4$.

## 5. Conclusions

Improvement in the performance of a production facility due to a learning effect is realistic and important. Solving scheduling problems with a learning effect requires more computational effort than the effort required for solving the original problems. This is verified even for the simplest classical single-machine makespan minimization problem. Moreover, we show that WSPT does not remain optimal for minimizing weighted flow-time, that EDD is not optimal for minimizing maximum lateness, and that Moore's algorithm does not produce the optimal schedule when minimizing the number of tardy jobs. We note that the complexity status of these problems remains an open question, and solving them seems an interesting topic for future research. We solve two multi-criteria problems that can be formulated as assignment problems. Finally, we show that SPT does not remain optimal for the minimum flow-time problem on parallel identical machines. It is suggested for future research to investigate the learning effect in the context of additional scheduling problems, including multi-machine and job-shops settings.

## Acknowledgements

This paper was supported in part by the Recanati Fund of the School of Business Administration, The Hebrew University, Jerusalem, Israel.

## References

Bagchi, U.B., 1989. Simultaneous minimization of mean and variation of flow-time and waiting time in single machine systems. Operations Research 37, 118–125.

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

Jackson, J.R., 1955. Scheduling a production line to minimize maximum tardiness. Research Report 43, Management Science Research Project, University of California, Los Angeles.

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

Nadler, G., Smith, W.D., 1963. Manufacturing progress functions for types of processes. International Journal of Production Research 2, 115–135.

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

Smith, M.L., 1956. Various optimizers for single-machine production. Naval Research Logistics Quarterly 3, 59–66.

Yelle, L.E., 1979. The learning curve: Historic review and comprehensive survey. Decision Science 10, 302–328.