# Methods for scheduling problems considering experience, learning and forgetting effects

Xiaoping Li, *Senior Member, IEEE,*, Yulu Jiang and Rubén Ruiz

***Abstract*—Workers with different levels of experience and knowledge have different effects on job processing times. By taking into account three factors: the sum-of-processing-time, the job-position, and the experience of workers, a more general learning model is introduced for scheduling problems. We show that this model generalizes existing ones and brings the consideration of learning and forgetting effects closer to reality. We demonstrate that some single machine scheduling problems are polynomially solvable under this general model. Considering the forgetting effect caused by the idle time on the second machine, we construct a learning-forgetting model for the two-machine permutation flow shop scheduling problem with makespan minimization. A branch and bound method and four heuristics are presented to find optimal and approximate solutions respectively. The proposed heuristics are evaluated over a large number of randomly generated instances. Experimental results show that the proposed heuristics are effective and efficient.**

***Index Terms*—Flowshop, Learning effect, Forgetting effect, Experience**

## I. Introduction

In classical scheduling, the processing time of a job is assumed to be known in advance and deterministic. However, in many realistic situations, the production facility (i.e. a worker) gains knowledge or experience by repeating similar production operations which improves production efficiency. Therefore, if a job is processed later in the sequence, its actual processing time is shorter. This phenomenon is known as the "learning effect" and was first studied by Badiru [1].

There are two fundamental learning effect models according to Biskup [2]: the position-based learning effect and the sum-of-processing-times-based learning effect. The typical position-based learning model $p_{jr}=p_j r^{\alpha}$ was proposed by Biskup [3], where $p_{jr}$ is the actual processing time of job $j$ scheduled at position $r$ of the sequence, $p_j$ is the normal processing time of job $j$, and $\alpha\leq 0$ is the learning index. Based on this model, Biskup [3] and Mosheiov [4] used the pairwise interchange technique and proved that the shortest processing time (SPT) rule could obtain the optimal sequence for single machine scheduling problems to minimize total completion time and makespan respectively. Zhao et al. [5] showed that the single machine scheduling minimizing the total weighted completion time and maximum lateness objectives are polynomially solvable problems. This was done by sorting jobs using the weighted shortest processing time (WSPT) rule and the earliest due date (EDD) rule under certain agreeable conditions. Based on the sum of processing times, Kuo & Yang [6] developed a sum-of-processing-time-based learning model $p_{jr}=p_j(1+\sum_{k=1}^{r-1}p_{[k]})^{\alpha}$, where $p_{[k]}$ is the normal processing time of a job scheduled at position $k$ in a sequence. In addition, some other time-based learning effect models [7], [8], [9], [10], [11], [12] were constructed in terms of the two fundamental ones.

Taking into account some factors, different learning effect models have been proposed for scheduling problems since Biskup [3] and Cheng & Wang [13] introduced learning effects into this field. Janiak & Rudel [14] investigated an experience-based learning effect model in which the learning curve is S-shaped. Later, they presented the multi-ability learning model [15] that generalizes the previous ones. This model is closer to real-life settings than the existing ones. For some special cases in the makespan minimization problems, they proposed optimal algorithms with polynomial times. Wang et al. [16] investigated some single-machine problems with past-sequence-dependent setup times while Lee et al. [17] focused on a single-machine problem with the learning effect and release times to minimize makespan. Yang et al. [18] introduced a new learning effect model, which is sum-of-processing-time-based, job-position-based and job complexity based. Wu et al. [19] and Cheng et al. [20] considered two-machine flowshop problems with a truncated learning function where the actual job processing time is related to a learning truncation parameter. Truncation refers to a limited learning effect and total completion time and makespan are respectively minimized in their problems.

Besides the learning effect models with specific functions, some general models have been investigated. Lai & Lee [21] proposed the general learning effect model $p^A_{j[r]}=p_j f(\sum_{k=1}^{r-1}\beta_k p_{[k]},r)$. The sum of the normal processing times of scheduled or processed jobs and the current position are the two variables of the learning function. Different learning curves can be constructed easily according to this model. Many existing models are special cases of this general model. Based on the model proposed by Yin et al. [11], Wang et al. [22] introduced the general model $p^A_{j[r]}=p_j f(\sum_{k=1}^{r-1}\beta_k p_{[k]})g(r)$, which is similar to that given in [21]. Recently, Lai & Lee [23] considered the forgetting effect and developed a general learning-forgetting effect model $p^A_{j[r]}=p_j f(\sum_{k=1}^{r-1}\beta_k p_{[k]},r)g(r)$ $(r>m)$, where $g(r)$ is the forgetting effect function. The works on the forgetting effect are less numerous than those on the

Xiaoping Li and Yulu Jiang are with the School of Computer Science and Engineering, Southeast University, Nanjing 211189, China, and also with Key Laboratory of Computer Network and Information Integration, Ministry of Education, Nanjing, 211189, China (e-mail: xpli@seu.edu.cn).

Rubén Ruiz is with Grupo de Sistemas de Optimización Aplicada, Instituto Tecnológico de Informática, Ciudad Politécnica de la Innovación, Edifico 8G, Acc. B. Universitat Politècnica de València, Camino de Vera s/n, 46021, València, Spain (e-mail: rruiz@eio.upv.es).

learning effect. Yang & Chand [24] studied a group scheduling problem with learning and forgetting effects. They focused on the learning effect of jobs in the same family and the forgetting effect among families. In addition, the deteriorating effect was viewed as a general forgetting effect, i.e., the processing ability of the production facility deteriorates with the waiting length for processing which makes the actual processing time possibly longer than the normal one. When considered together, learning and forgetting effects can be used to model a myriad of situations in practice and are much more general than fixed processing times. For example, when a production line has been producing a product for a while, not only workers have learned to be more efficient but also intermediate storage is full, production cells are prepared with all needed materials, auxiliary tooling is prepared and in working order, etc. This all results in more efficient production and higher efficiency (shorter production times). When a different product is produced, all these activities have to be restarted and the learning is lost. Similarly, if a type of product has not been produced for a number of shifts, all this tooling and preparation is lost and hence the forgetting effect.

In realistic production systems, experience (or prior knowledge) exerts a great impact on the learning effect from the perspective of cognitive psychology [25], [26], [27]. For instance, workers in factories are usually classified into junior, intermediate and senior ones according to their experience. They are assigned to different positions with distinct skill requirements to increase productivity.

A clear example is a shoe manufacturing production shop. The main stages are upper shoe manufacturing (which typically involves cutting leather, skiving, crimping, stitching and attaching although this varies wildly from product to product), insole manufacturing (attachment of leather sole, shank and half sole), lasting and bottoming (with many steps, including fitting the shoe upper, ankle stiffening, stitching insole and attaching sole, etc.) and finally finishing and packaging. It is easy to see that different levels of experience among the workers would result in different processing speeds for each manufactured shoe (and also, different quality results). Similarly, when a worker has been doing the same repetitive upper shoe leather cutting, the step is faster and faster each time. This is the learning effect. If after cutting the same type of leather for a specific type of shoe the worker makes a short production run and cuts leather for another shoe type, he/she starts forgetting the previous shoe type. The longer the time before going back to the initial shoe type, the stronger the forgetting effect. In a extreme case, if the worker goes back to the initial shoe type after a week of production, he/she will have a very slow start again. This is the forgetting effect. As these effects end up having a sizable impact on the processing times and these times do depend on the processing sequence, they also have a big impact on the makespan objective.

Inspired by this, we introduce a general learning effect model with experience, and analyze its properties for some single machine problems. For two- or multiple-machine permutation flow shop problems, there might be some gaps or idle times between consecutive operations. These gaps cause forgetting effects by interrupting the learning process and reduce learning effects. We consider the two-machine permutation flow shop problem with learning effects involving both experience and forgetting effects to minimize makespan. Based on the introduced general learning effect model, two specific learning-forgetting effect models with experience are developed. Four two-phase heuristic algorithms are proposed to obtain near-optimal solutions. Six lower bounds are obtained for speeding up the elimination process of a proposed branch and bound algorithm. Therefore, the main contributions of this paper are the following:

- A generalization of the learning effects model with learning, experience and forgetting effects is proposed.
- Some polynomially solvable solvable cases for single machines are identified.
- A branch and bound procedure, based on six tight lower bounds is proposed for the two machine flowshop problem.
- Four heuristics are further presented for the same two machine problem.
- Detailed and comprehensive computational experiments, along with statistical analyses, show the performance of the proposed methods.

The remainder of this paper is organized as follows: The general learning effect model with experience is given in Section II. Section III provides polynomial-time solution algorithms for some single-machine problems. A learning-forgetting effect model, four heuristic algorithms, six lower bounds and a branch and bound method are developed to solve two-machine permutation flow shop problems in Section IV. Section V gives the experimental results, followed by conclusions and future research in Section VI.

## II. General learning effect model with experience

A learning effect model considering experience is closer to real-life applications, in which the following aspects are usually assumed:

(i) The actual job processing time of a worker with experience is less than that without experience. For example, a novice without experience spends 10 units of time processing a job while he/she needs only 8 units to finish the same job if he/she has some experience.

(ii) A bigger experience factor implies more learning effects for the same worker. For example, a worker with the experience factor 0.1 spends 8 time units processing a job while the actual job processing time would be 6 time units if the experience factor is 0.2. Obviously, the learning effect model with the experience factor 0 should be identical to that without experience.

The learning effects result in the decrease of actual processing times. However, the actual processing time would decrease little when the learning effect reaches a given threshold. In terms of the model $p_{jr}=p_j f(\sum_{k=1}^{r-1}\beta_k p_{[k]})g(r)$ proposed in [28],

we design a general experience learning effect model as follows:

$$p_{jr}=p_j\max\Big\{(1-\omega)f(\sum_{k=1}^{r-1}\beta_k p_{[k]})g(r),\theta\Big\} \quad (1)$$

where $p_{jr}$ is the actual processing time of job $j$ scheduled at position $r$, $p_j$ is the normal processing time of job $j$, $\omega$ ($0\leq\omega<1$) denotes the experience factor, $\theta$ ($0\leq\theta<1$) is the threshold. $f$:$[0,+\infty)\rightarrow(0,1]$ is a differentiable non-increasing function so that $f'$ is non-decreasing in $[0,+\infty)$ and $f(0)=1$. $g$:$[1,+\infty)\rightarrow(0,1]$ is a non-increasing function with $g(1)=1$, and $\beta_r$ is the weight on position $r$. We assume that $0\leq\beta_1\leq\beta_2\leq...\leq\beta_n$.

Kuo & Yang [6] developed a sum-of-processing-time-based learning model $p_{jr}=p_j(1+\sum_{k=1}^{r-1}p_{[k]})^{a}$. Yin et al. [11] generalized it to a position-dependent and time-dependent learning effect model $p_{jr}=p_j f(\sum_{k=1}^{r-1}p_{[k]})g(r)$, which was extended to

$$p_{jr}=p_j f(\sum_{k=1}^{r-1}\beta_k p_{[k]})g(r) \quad (2)$$

with weights $0\leq\beta_1\leq\beta_2\leq...\leq\beta_n$ by Wang et al. [28]. Additionally, Cheng et al. [29] observed that the actual processing time of a job under an uncontrolled learning effect would drop to zero precipitously as the number of jobs already processed increases or jobs with long processing times exist. They developed the learning effect model

$$p_{jr}=p_j\max\Big\{\Big(1+\sum_{k=1}^{r-1}p_{[k]}\Big)^{\alpha},\theta\Big\} \quad (3)$$

where $\alpha<0$ is the learning index and $0<\theta<1$ is the truncation parameter. Therefore, the designed experience learning effect model proposed in Equation (1) is more general than those in [28] and [29].

Equation (2) is a generalization of the models developed in [6] and [11] but it has to be noted that the actual processing time of a job was in those previous works not controlled by a threshold $\theta$ ($0<\theta<1$). This means that zero processing might have resulted which is impossible in practical settings. On the other hand, even though the actual processing time of a job is limited by a threshold $\theta$ ($0<\theta<1$) in Equation (3), the function $p_j\Big(1+\sum_{k=1}^{r-1}p_{[k]}\Big)^{\alpha}$ is identical to that of [6] which at the same time it is a special case of Equation (2). In addition, the experience factor $\omega$ ($0\leq\omega<1$) exerts an important influence on the actual processing times which has never been considered in the scheduling literature yet. Therefore, the newly designed model in Equation (1) becomes Equation (2) when $\omega=0$ and $\theta=0$. In other words, the designed model is identical to the model developed by Wang et al. [28] if neither experience nor threshold are considered. When $\omega=0$, $g(r)=1$, $\beta_1=\beta_2=...=\beta_n=1$, Equation (1) turns into $p_{jr}=p_j\max\Big\{f(\sum_{k=1}^{r-1}p_{[k]}),\theta\Big\}$. Furthermore, if we take $f(\sum_{k=1}^{r-1}p_{[k]})=\Big(1+\sum_{k=1}^{r-1}p_{[k]}\Big)^{\alpha}$ and $0<\theta<1$, the model becomes Equation (3). Again, the proposed model is identical to that introduced by Cheng et al. [29] if we just consider the specific learning effect model $\Big(1+\sum_{k=1}^{r-1}p_{[k]}\Big)^{\alpha}$ without dealing with experience. The conclusion is that we are presenting a more general and flexible model that captures more real-life scenarios than the previous ones.

## III. Single-machine problems

We prove the following theorems using the pairwise interchange technique. Suppose that $\pi$ and $\pi'$ are two job schedules and the difference between $\pi$ and $\pi'$ is a pairwise interchange of two adjacent jobs $j$ and $k$. As shown in Fig.1, $\pi=[S_1,j,k,S_2]$ and $\pi'=[S_1,k,j,S_2]$, where $S_1$ and $S_2$ are partial sequences. Furthermore, we assume that there are $r-1$ scheduled jobs in $S_1$, $B$ is the completion time of the last job in $S_1$, $A=\sum_{i=1}^{r-1}\beta_i p_{[i]}$ and $h$ is the first job in $S_2$.

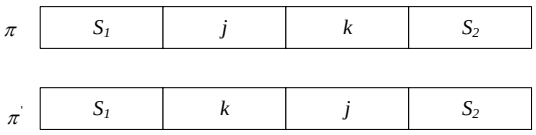


Fig. 1. A pairwise interchange of adjacent jobs

The completion times of jobs $j$ and $k$ in $\pi$ are

$$C_j(\pi)=B+p_j\max\{(1-\omega)f(A)g(r),\theta\} \quad (4)$$

and

$$\begin{aligned}C_k(\pi)=\ & B+p_j\max\{(1-\omega)f(A)g(r),\theta\}\\ &+p_k\max\{(1-\omega)f(A+\beta_r p_j)g(r+1),\theta\}\end{aligned} \quad (5)$$

Similarly, the completion times of jobs $k$ and $j$ in $\pi'$ are

$$C_k(\pi')=B+p_k\max\{(1-\omega)f(A)g(r),\theta\} \quad (6)$$

and

$$\begin{aligned}C_j(\pi')=\ & B+p_k\max\{(1-\omega)f(A)g(r),\theta\}\\ &+p_j\max\{(1-\omega)f(A+\beta_r p_k)g(r+1),\theta\}\end{aligned} \quad (7)$$

**Theorem 1** *For the problem* $1|p_{jr}=p_j\max\Big\{(1-\omega)f(\sum_{i=1}^{r-1}\beta_i p_{[i]})g(r),\theta\Big\}|C_{\max}$, *an optimal schedule can be obtained by sequencing the jobs in non-decreasing order of* $p_j$ *(the SPT rule).* □

**Proof 1** *Suppose that* $p_j\leq p_k$. *To show that* $\pi$ *dominates* $\pi'$, *it suffices to show that* $C_k(\pi)\leq C_j(\pi')$ *and* $C_u(\pi)\leq C_u(\pi')$ *for any job* $u$ *in* $S_2$. *We prove them respectively in the following.*

*First, we prove that* $C_k(\pi)\leq C_j(\pi')$. *Regarding the difference between Equations (5) and (7), we have*

$$\begin{aligned}C_j(\pi')-C_k(\pi)=\ & (p_k-p_j)\max\{(1-\omega)f(A)g(r),\theta\}+\\ & p_j\max\{(1-\omega)f(A+\beta_r p_k)g(r+1),\theta\}\\ & -p_k\max\{(1-\omega)f(A+\beta_r p_j)g(r+1),\theta\}\end{aligned} \quad (8)$$

$f$ *and* $g$ *are non-increasing functions which demonstrate that* $(1-\omega)f(A+\beta_r p_k)g(r+1)\leq(1-\omega)f(A+\beta_r p_j)g(r+1)\leq(1-\omega)f(A)g(r)$. *There are four cases for the parameter* $\theta$ *of Equation* (8)*:*

*(i) If* $(1-\omega)f(A+\beta_r p_k)g(r+1)\leq(1-\omega)f(A+\beta_r p_j)g(r+1)\leq(1-\omega)f(A)g(r)\leq\theta$,

$$C_j(\pi')-C_k(\pi)=(p_k-p_j)\theta+p_j\theta-p_k\theta=0$$

*(ii) If* $(1-\omega)f(A+\beta_r p_k)g(r+1)\leq(1-\omega)f(A+\beta_r p_j)g(r+1)\leq\theta\leq(1-\omega)f(A)g(r)$,

$$\begin{aligned}C_j(\pi')-C_k(\pi)&=(p_k-p_j)(1-\omega)f(A)g(r)+p_j\theta-p_k\theta\\ &=(p_k-p_j)[(1-\omega)f(A)g(r)-\theta]\geq 0\end{aligned}$$

*(iii) If* $\theta\leq(1-\omega)f(A+\beta_r p_k)g(r+1)\leq(1-\omega)f(A+\beta_r p_j)g(r+$

$1)\leq(1-\omega)f(A)g(r)$,

$$\begin{aligned}C_j(\pi')-C_k(\pi)=&\ (p_k-p_j)(1-\omega)f(A)g(r)+p_j(1-\omega)\times\\ &f(A+\beta_r p_k)g(r+1)-p_k(1-\omega)f(A+\beta_r p_j)g(r+1)\\ =&\ (1-\omega)[(p_k-p_j)f(A)g(r)+p_j f(A+\beta_r p_k)g(r+1)\\ &-p_k f(A+\beta_r p_j)g(r+1)]\end{aligned}$$

*According to the proof of Theorem 1 in [28],* $(p_k-p_j)f(A)g(r)+p_j f(A+\beta_r p_k)g(r+1)-p_k f(A+\beta_r p_j)g(r+1)\geq 0$. *Based on the assumption* $0\leq\omega<1$, *we have* $C_j(\pi')-C_k(\pi)\geq 0$.

*(iv) If* $(1-\omega)f(A+\beta_r p_k)g(r+1)\leq\theta\leq(1-\omega)f(A+\beta_r p_j)g(r+1)\leq(1-\omega)f(A)g(r)$,

$$\begin{aligned}C_j(\pi')-C_k(\pi)=&\ (p_k-p_j)(1-\omega)f(A)g(r)+p_j\theta\\ &-p_k(1-\omega)f(A+\beta_r p_j)g(r+1)\end{aligned}$$

*Since* $\theta\geq(1-\omega)f(A+\beta_r p_k)g(r+1)$,

$$\begin{aligned}C_j(\pi')-C_k(\pi)\geq&\ (p_k-p_j)(1-\omega)f(A)g(r)\\ &+p_j(1-\omega)f(A+\beta_r p_k)g(r+1)\\ &-p_k(1-\omega)f(A+\beta_r p_j)g(r+1)\end{aligned}$$

*Similarly to case (iii), we have* $C_j(\pi')-C_k(\pi)\geq 0$.

*The above four cases imply that* $C_j(\pi')-C_k(\pi)\geq 0$ *for any* $\theta$.

*Secondly,* $C_h(\pi)=C_k(\pi)+p_h\max\{(1-\omega)f(A+\beta_r p_j+\beta_{r+1}p_k)g(r+2),\theta\}$ *and* $C_h(\pi')=C_j(\pi')+p_h\max\{(1-\omega)f(A+\beta_r p_k+\beta_{r+1}p_j)g(r+2),\ \theta\}$. *Since* $p_j\leq p_k$ *and* $\beta_r\leq\beta_{r+1}$, *we have* $\beta_r p_j+\beta_{r+1}p_k-\beta_r p_k-\beta_{r+1}p_j=(p_k-p_j)(\beta_{r+1}-\beta_r)\geq 0$. *The fact that* $f$ *and* $g$ *are non-increasing functions implies that* $(1-\omega)f(A+\beta_r p_j+\beta_{r+1}p_k)g(r+2)\leq(1-\omega)f(A+\beta_r p_k+\beta_{r+1}p_j)g(r+2)$. *Similar to the above proof process, we can obtain* $C_h(\pi')\geq C_h(\pi)$ *and* $C_u(\pi')\geq C_u(\pi)$ *for any job* $u$ *in* $S_2$.

*All in all,* $\pi$ *dominates* $\pi'$. *An optimal schedule can be obtained by sequencing jobs in non-decreasing order of* $p_j$. □

**Theorem 2** *For the problem* $1|p_{jr}=p_j\max\Big\{(1-\omega)f(\sum_{i=1}^{r-1}\beta_i p_{[i]})g(r),\theta\Big\}|\sum w_jC_j$, *if the jobs have agreeable weights, i.e.,* $p_j\leq p_k$ *implies* $w_j\geq w_k$ *for all the jobs* $j$ *and* $k$, *then an optimal schedule can be obtained by sequencing the jobs in non-decreasing order of* $\frac{p_j}{w_j}$ *(i.e., the WSPT rule).* □

**Proof 2** *Suppose that* $p_j\leq p_k$. *According to Theorem 1,* $C_k(\pi)\leq C_j(\pi')$. *To show that* $\pi$ *dominates* $\pi'$, *it suffices to show that* $\sum_{j=1}^n w_jC_j(\pi)\leq\sum_{j=1}^n w_jC_j(\pi')$.

*Obviously, the completion times of the same job* $i$ *in* $S_1$ *of sequences* $\pi$ *and* $\pi'$ *are equal and* $\sum_{i=1}^{r-1}w_{[i]}C_{[i]}(\pi)=\sum_{i=1}^{r-1}w_{[i]}C_{[i]}(\pi')$. *According to Equations (5) and (7),*

$$\begin{aligned}&w_kC_k(\pi')+w_jC_j(\pi')-w_jC_j(\pi)-w_kC_k(\pi)\\ =&(w_j+w_k)(p_k-p_j)\max\{(1-\omega)f(A)g(r),\theta\}+w_jp_j\max\{(1-\omega)\\ &f(A+\beta_rp_k)g(r+1),\theta\}-w_kp_k\max\{(1-\omega)f(A+\beta_rp_j)g(r+1),\theta\}\end{aligned}$$

*Now we prove that* $w_kC_k(\pi')+w_jC_j(\pi')\geq w_jC_j(\pi)+w_kC_k(\pi)$ *for any* $\theta$.

*Since* $f$ *and* $g$ *are non-increasing functions,* $(1-\omega)f(A+\beta_rp_k)g(r+1)\leq(1-\omega)f(A+\beta_rp_j)g(r+1)\leq(1-\omega)f(A)g(r)$. *Similar to the proof process for Theorem 1, there are four cases for the parameter* $\theta$:

*(i)* $(1-\omega)f(A+\beta_rp_k)g(r+1)\leq(1-\omega)f(A+\beta_rp_j)g(r+1)\leq(1-\omega)f(A)g(r)\leq\theta$

$$\begin{aligned}&w_kC_k(\pi')+w_jC_j(\pi')-w_jC_j(\pi)-w_kC_k(\pi)\\ =&(w_j+w_k)(p_k-p_j)\theta+w_jp_j\theta-w_kp_k\theta\\ =&(w_jp_k-w_kp_j)\theta\end{aligned}$$

$p_j\leq p_k$ *and* $w_j\geq w_k$ *imply that* $(w_jp_k-w_kp_j)\theta\geq 0$. *Then* $w_kC_k(\pi')+w_jC_j(\pi')\geq w_jC_j(\pi)+w_kC_k(\pi)$.

*(ii)* $(1-\omega)f(A+\beta_rp_k)g(r+1)\leq(1-\omega)f(A+\beta_rp_j)g(r+1)\leq\theta\leq(1-\omega)f(A)g(r)$

$$\begin{aligned}&w_kC_k(\pi')+w_jC_j(\pi')-w_jC_j(\pi)-w_kC_k(\pi)\\ =&(w_j+w_k)(p_k-p_j)(1-\omega)f(A)g(r)+w_jp_j\theta-w_kp_k\theta\end{aligned}$$

$(1-\omega)f(A)g(r)\geq\theta$ *illustrates that* $w_kC_k(\pi')+w_jC_j(\pi')-w_jC_j(\pi)-w_kC_k(\pi)\geq(w_j+w_k)(p_k-p_j)\theta+w_jp_j\theta-w_kp_k\theta=(w_jp_k-w_kp_j)\theta$.

*Similar to the case (i), we have* $(w_jp_k-w_kp_j)\theta\geq 0$, *i.e.,* $w_kC_k(\pi')+w_jC_j(\pi')\geq w_jC_j(\pi)+w_kC_k(\pi)$.

*(iii)* $\theta\leq(1-\omega)f(A+\beta_rp_k)g(r+1)\leq(1-\omega)f(A+\beta_rp_j)g(r+1)\leq(1-\omega)f(A)g(r)$

$$\begin{aligned}&w_kC_k(\pi')+w_jC_j(\pi')-w_jC_j(\pi)-w_kC_k(\pi)\\ =&(w_j+w_k)(p_k-p_j)(1-\omega)f(A)g(r)+w_jp_j(1-\omega)\times\\ &f(A+\beta_rp_k)g(r+1)-w_kp_k(1-\omega)f(A+\beta_rp_j)g(r+1)\\ =&(1-\omega)(w_j+w_k)p_jg(r)\Big[\Big(\frac{p_k}{p_j}-1\Big)f(A)+\frac{w_j}{w_j+w_k}\\ &f(A+\beta_rp_k)\frac{g(r+1)}{g(r)}-\frac{w_k}{w_j+w_k}\frac{p_k}{p_j}f(A+\beta_rp_j)\frac{g(r+1)}{g(r)}\Big]\end{aligned}$$

*Assuming* $\delta_1=\frac{w_k}{w_j+w_k}\times\frac{g(r+1)}{g(r)}$, $\delta_2=\frac{w_j}{w_j+w_k}\times\frac{g(r+1)}{g(r)}$, $\chi=\beta_rp_j$, $\lambda=\frac{p_k}{p_j}$. *Since* $w_j\geq w_k$, $\delta_2\geq\delta_1$. *Therefore*

$$\begin{aligned}&\frac{w_kC_k(\pi')+w_jC_j(\pi')-w_jC_j(\pi)-w_kC_k(\pi)}{(1-\omega)(w_j+w_k)p_jg(r)}\\ =&(\lambda-1)f(A)+\delta_2f(A+\lambda\chi)-\delta_1f(A+\chi)\\ \geq&(\lambda-1)f(A)+\delta_1f(A+\lambda\chi)-\delta_1f(A+\chi)\end{aligned}$$

*From the proof of Lemma 1 in [28], we have* $(\lambda-1)f(A)+\delta_1f(A+\lambda\chi)-\delta_1f(A+\chi)\geq 0$. *So* $w_kC_k(\pi')+w_jC_j(\pi')\geq w_jC_j(\pi)+w_kC_k(\pi)$.

*(iv)* $(1-\omega)f(A+\beta_rp_k)g(r+1)\leq\theta\leq(1-\omega)f(A+\beta_rp_j)g(r+1)\leq(1-\omega)f(A)g(r)$

$$\begin{aligned}&w_kC_k(\pi')+w_jC_j(\pi')-w_jC_j(\pi)-w_kC_k(\pi)\\ =&(w_j+w_k)(p_k-p_j)(1-\omega)f(A)g(r)+w_jp_j\theta\\ &-w_kp_k(1-\omega)f(A+\beta_rp_j)g(r+1)\end{aligned}$$

*Since* $\theta\geq(1-\omega)f(A+\beta_rp_k)g(r+1)$, *then*

$$\begin{aligned}&w_kC_k(\pi')+w_jC_j(\pi')-w_jC_j(\pi)-w_kC_k(\pi)\\ >&(w_j+w_k)(p_k-p_j)(1-\omega)f(A)g(r)\\ &+w_jp_j(1-\omega)f(A+\beta_rp_k)g(r+1)\\ &-w_kp_k(1-\omega)f(A+\beta_rp_j)g(r+1)\end{aligned}$$

*Similar to the case (iii), we have* $w_kC_k(\pi')+w_jC_j(\pi')\geq w_jC_j(\pi)+w_kC_k(\pi)$.

*Therefore,* $w_kC_k(\pi')+w_jC_j(\pi')\geq w_jC_j(\pi)+w_kC_k(\pi)$. *From the proof of Theorem 1, we have* $C_u(\pi')\geq C_u(\pi)$ *for any job* $u$ *in* $S_2$. *So* $\sum_{j=1}^n w_jC_j(\pi')\geq\sum_{j=1}^n w_jC_j(\pi)$. *In other words,* $\pi$ *dominates* $\pi'$. *An optimal schedule can be obtained by sequencing the jobs in non-decreasing order of* $\frac{p_j}{w_j}$ *or repeating this interchange procedure to all unsequenced jobs using the WSPT rule.* □

A special case of Theorem 2 is the following:

**Corollary 1** *For the problem* $1|p_{jr}=p_j\max\left\{(1-\omega)f(\sum_{i=1}^{r-1}\beta_i p_{[i]})g(r),\theta\right\}|\sum C_j$, *an optimal schedule can be obtained by sequencing the jobs in non-decreasing order of* $p_j$ *(the SPT rule).* □

Similarly to the Theorem 7 in [28] and the Theorem 2.3 in [12], we have

**Theorem 3** *For the problem* $1|p_{jr}=p_j\max\left\{(1-\omega)f(\sum_{i=1}^{r-1}\beta_i p_{[i]})g(r),\theta\right\}|L_{\max}$, *if the job's processing times and due dates are agreeable, i.e.,* $d_j\le d_k$ *implies* $p_j\le p_k$ *for all the jobs* $j$ *and* $k$, *then an optimal schedule can be obtained by sequencing the jobs in non-decreasing order of the due date* $d_j$ *(the EDD rule).* □

**Theorem 4** *For the problem* $1|p_{jr}=p_j\max\left\{(1-\omega)f(\sum_{i=1}^{r-1}\beta_i p_{[i]})g(r),\theta\right\}|\sum T_j$, *if the job's processing times and due dates are agreeable, i.e.,* $d_j\le d_k$ *implies* $p_j\le p_k$ *for all the jobs* $j$ *and* $k$, *then an optimal schedule can be obtained by sequencing the jobs in non-decreasing order of the due date* $d_j$ *(the EDD rule).* □

For an example, we assume that 5 pairs of shoes to be leather cut by a worker (especially when he/she shifts to this operation). Their normal processing times are 30, 46, 28, 50 and 35 respectively. We assume that $f(\sum_{k=1}^{r-1}p_{[k]})=(1+\sum_{k=1}^{r-1}p_{[k]})^{\alpha}$, $g(r)=1$, $\omega=0$, $\alpha=-0.1$, $\beta_1=\beta_2=\beta_3=\beta_4=\beta_5=1$, $\theta=0.6$. Using the SPT rule on normal processing times, the schedule (3, 1, 5, 2, 4) is obtained for the problem $1|p_{jr}=p_j\max\left\{(1-\omega)f(\sum_{i=1}^{r-1}\beta_i p_{[i]})g(r),\theta\right\}|C_{\max}$. According to equation (1), actual processing times of the five operations are 21.42, 29.91, 28.00, 31.43 and 23.65 respectively. Makespan of the schedule is 134.41 and that would be 189 without learning and forgetting effects.

## IV. The Two-Machine Permutation Flow Shop Problem

A permutation flow shop (PFSP) is a classic scheduling problem. Given a set of jobs $\mathbb{J}=\{1,2,...,n\}$ and a set of $m$ machines, each job has to be processed on all $m$ machines in the same order. The processing times of each job on each machine are given. At any time, each machine can process at most one job and each job can be processed at most on one machine. Interrupting the processing of any job on a machine is not permitted. All jobs are processed in the same order at every machine. In this paper, we consider the two-machine permutation flow shop problem minimizing the makespan with learning effects. For each job $j$, $a_j$ and $b_j$ denote the normal processing times on machine 1 and machine 2 respectively.

### A. Specific learning-forgetting models with experience

In the PFSP, operations on the first machine usually start as soon as possible, i.e., there is no gap between operations so no forgetting effect is considered. Similarly to [28], we specify the proposed general learning effect model for operations on machine 1 with $f(\sum_{i=1}^{r-1}\beta_i p_{[i]})=\left(1-\frac{\sum_{i=1}^{r-1}p_{[i]}}{\sum_{i=1}^{n}p_i}\right)^{\alpha_1}$, $g(r)=r^{\alpha_2}$, $\alpha_1\ge 1$, $\alpha_2<0$. Taking into account all the parameters (learning, position, threshold and experience), we define $S_{[x,k]}=(1-\omega)\left(1-\frac{\sum_{l=1}^{k}x_{[l]}}{\sum_{l=1}^{n}x_{[l]}}\right)^{\alpha_1}(k+1)^{\alpha_2}$ where $x\in\{a,b\}$. $L(x,k,\theta)=\max\left\{S_{[x,k]},\theta\right\}$. Because there is no idle time on machine 1, no forgetting is involved on the this machine. The specific learning model with experience on machine 1 is

$$a_{jr}=a_j\max\left\{(1-\omega)\left(1-\frac{\sum_{k=1}^{r-1}a_{[k]}}{\sum_{k=1}^{n}a_k}\right)^{\alpha_1}r^{\alpha_2},\theta\right\}=a_jL(a,r-1,\theta) \tag{9}$$

The actual processing time of job $j$ at the $r^{th}$ position on machine 2 would be $b_{jr}=b_jL(b,r-1,\theta)$ if there were no idle time. However, idle times might appear on machine 2 leading to forgetting effects, which always decrease what has been learned in the learning effects. Forgetting effects are closely related to the total amount of what has been learned and the sum duration of the breaks before the current position $r$. Inspired by [30] and [31], we design the specific learning-forgetting model with experience on machine 2, in which the more learning there is the less forgetting. In addition, the longer the sum duration of the breaks is, the more forgetting there is. The model is formulated as follows:

$$b_{jr}=b_jL(b,r-1,\theta)+b_j(1-L(b,r-1,\theta))(1-e^{-\sigma\sum_{k=1}^{r}I_{[k]}}) \tag{10}$$

where $I_{[k]}$ is the idle time of a job scheduled at position $k$, $\sigma(\sigma<0)$ is the forgetting index, $b_j(1-L(b,r-1,\theta))$ is the amount learned.

### B. Heuristic algorithms

Though the Johnson Rule obtains the optimum solution for the classical two-machine permutation flow shop scheduling problem with makespan minimization [32], it fails to optimally solve the considered two-machine problem with learning effects because the processing times are not determined in advance. Cheng et al. [20] considered the two-machine flowshop problem with a truncated learning function $a_{jr}=a_j\max\{r^{\alpha},\beta\}$ and $b_{jr}=b_j\max\{r^{\alpha},\beta\}$, of which the complexity is unknown. Obviously, the learning-forgetting models with experience developed in this paper are much more general than those in [20], i.e., the considered problem is more complex than that in [20]. Cheng et al. proposed a branch-and-bound and three crossover-based genetic algorithms (GAs) for a two-machine flowshop scheduling with a truncated learning function to minimize the makespan, which drives us to propose heuristics for the considered problem.

In this paper, a two-stage algorithm framework is proposed. There are two phases in the framework: the Initial Solution Construction and the Solution Improvement. The construction phase has two variants: the Johnson rule and the Greedy rule. The improvement phase contains two variants: the insertion policy or swap policy. Based on the variants, four heuristic algorithms are developed: JIH (Johnson And Insert), JSH (Johnson And Swap), GIH (Greedy And Insert) and GSH (Greedy And Swap). The components of the heuristics are described in Algorithms 1∼4.

In Step 10 of the Greedy Rule, the job with the minimum $a_j-b_{[k]}$ is selected to reduce the idle time of the second machine. For a fixed $b_{[k]}$, the job with the minimum of $a_j$ is selected from the remaining jobs in the set. This situation is

**Algorithm 1:** Johnson Rule

**begin**
  Construct a set of jobs $\mathbb{J}_1=\{j|j\in\mathbb{J}$ with $a_j\leq b_j\}$;
  Schedule jobs of $\mathbb{J}_1$ with the non-decreasing order of $a_j$, breaking ties arbitrarily;
  Schedule jobs of $\mathbb{J}\backslash\mathbb{J}_1$ with the non-increasing order of $b_j$, breaking ties arbitrarily;
  Construct $\pi_0$ by joining $\mathbb{J}_1$ followed by $\mathbb{J}\backslash\mathbb{J}_1$;
  **return** $\pi_0$

**Algorithm 2:** Greedy Rule

**begin**
  Construct an empty job sequence $S$;
  Construct a set of jobs $\mathbb{J}_1=\{j\in\mathbb{J}|a_j\leq b_j\}$, $k\leftarrow 1$;
  Schedule jobs of $\mathbb{J}\backslash\mathbb{J}_1$ with the non-increasing order of $b_j$;
  **if** $\mathbb{J}_1\neq\varnothing$ **then**
    Choose job $j$ with the minimum $b_j$ from $\mathbb{J}_1$ as the first element of $S$;
  **else**
    Choose job $j$ with the minimum $a_j$ from $\mathbb{J}\backslash\mathbb{J}_1$ as the first element of $S$;
  Delete job $j$ from $\mathbb{J}$;
  Choose job $j$ with the minimum $a_j-b_{[k]}$ from $\mathbb{J}$ as the $(k+1)^{th}$ element of $S$;
  Delete job $j$ from $\mathbb{J}$, $k\leftarrow k+1$;
  **if** $\mathbb{J}\neq\varnothing$ **then**
    Go to Step 10;
  $\pi_0\leftarrow S$;
  **return** $\pi_0$

similar to the SPT rule, i.e., the job with the longest processing time is scheduled at the latest position, which is in accordance with the learning effects.

It is easy to derive the time complexity of each variant of the first phase which is $O(n\log n)$ while that of the second phase is $O(n^2)$. Therefore, time complexity of the four heuristics is $O(n^2)$.

### C. Branch and bound algorithm

Besides the above heuristics, we propose a branch and bound algorithm for small-scale two-machine permutation flow shop problems. Six lower bounds are developed to trim the searching tree and to accelerate the searching process.

*1) Lower bounds:* Let $\pi^P$ be a partial schedule with $k$ jobs determined during the Branch and Bound process. $\pi$ is a complete schedule in which the first part is $\pi^P$. $A$ and $B$ are the completion times of the last job on machine 1 and machine 2 respectively. $M_{[k]}=1-e^{-\sigma\sum_{l=1}^{k}I_{[l]}}$ represents a function of the idle time resulting from the partial schedule $\pi^P$. Obviously, $0\leq M_{[1]}...\leq M_{[k]}\leq M_{[n]}<1$. Therefore, the completion times of the $(k+1)^{th}$ job on machines 1 and 2 are

**Algorithm 3:** Insertion Policy

**Input**: An initial solution $\pi_0$
**begin**
  **for** $k=1$ **to** $n-1$ **do**
    **for** $i=k+1$ **to** $n$ **do**
      Construct sequence $\pi$ by removing the $i^{th}$ job from $\pi_0$ and inserting it to the $k^{th}$ slot;
      **if** $C_{\max}(\pi)<C_{\max}(\pi_0)$ **then**
        Replace $\pi_0$ with $\pi$;
  **return** $\pi_0$

**Algorithm 4:** Swap Policy

**Input**: An initial solution $\pi_0$
**begin**
  **for** $k=1$ **to** $n-1$ **do**
    **for** $i=k+1$ **to** $n$ **do**
      Construct sequence $\pi$ by swapping the $i^{th}$ and the $k^{th}$ jobs of $\pi_0$;
      **if** $C_{\max}(\pi)<C_{\max}(\pi_0)$ **then**
        Replace $\pi_0$ with $\pi$;
  **return** $\pi_0$

$$
\begin{aligned}
C_{1[k+1]}&=A+a_{[k+1]}\max\Big\{(1-\omega)\Big(1-\frac{\sum_{l=1}^{k}a_{[l]}}{\sum_{l=1}^{n}a_{[l]}}\Big)^{\alpha_1}(k+1)^{\alpha_2},\theta\Big\}\\
&\geq A+a_{[k+1]}S_{[a,k]}\\
C_{2[k+1]}&=\max\{C_{1[k+1]},B\}+b_{[k+1]}\max\Big\{(1-\omega)\times\\
&\quad\Big(1-\frac{\sum_{l=1}^{k}b_{[l]}}{\sum_{l=1}^{n}b_{[l]}}\Big)^{\alpha_1}(k+1)^{\alpha_2},\theta\Big\}+\Big[b_{[k+1]}-b_{[k+1]}\times\\
&\quad\max\Big\{(1-\omega)\Big(1-\frac{\sum_{l=1}^{k}b_{[l]}}{\sum_{l=1}^{n}b_{[l]}}\Big)^{\alpha_1}(k+1)^{\alpha_2},\theta\Big\}\Big]M_{[k+1]}\\
&=\max\{C_{1[k+1]},B\}+b_{[k+1]}\max\Big\{S_{[b,k]},\theta\Big\}+\Big[b_{[k+1]}-\\
&\quad b_{[k+1]}\max\Big\{S_{[b,k]},\theta\Big\}\Big]M_{[k+1]}\\
&=\max\{C_{1[k+1]},B\}+(1-M_{[k+1]})b_{[k+1]}\max\Big\{S_{[b,k]},\theta\Big\}+\\
&\quad b_{[k+1]}M_{[k+1]}\\
&\geq C_{1[k+1]}+(1-M_{[k+1]})b_{[k+1]}S_{[b,k]}+b_{[k+1]}M_{[k+1]}\\
&\geq A+a_{[k+1]}S_{[a,k]}+b_{[k+1]}\Big[S_{[b,k]}+M_{[k+1]}-M_{[k+1]}S_{[b,k]}\Big]
\end{aligned}
$$

Similarly, we define

$$Z_{[x,i,j]}=(1-\omega)\Big(1-\frac{\sum_{l=1}^{i}x_{[l]}+\sum_{l=1}^{j}x_{[k+l]}}{\sum_{l=1}^{n}x_{[l]}}\Big)^{\alpha_1}(i+j+1)^{\alpha_2}$$

where $x\in\{a,b\}$. For the last job on machines 1 and 2

$$
\begin{aligned}
C_{1[n]}&\geq A+\sum_{i=1}^{n-k}a_{[k+i]}(1-\omega)\Big(1-\frac{\sum_{l=1}^{k}a_{[l]}+\sum_{l=1}^{i-1}a_{[k+l]}}{\sum_{l=1}^{n}a_{[l]}}\Big)^{\alpha_1}(k+i)^{\alpha_2}\\
&=A+\sum_{i=1}^{n-k}a_{[k+i]}Z_{[a,k,i-1]} \qquad (11)\\
C_{2[n]}&\geq A+\sum_{i=1}^{n-k}a_{[k+i]}Z_{[a,k,i-1]}
\end{aligned}
$$

$$+b_{[n]}(1-\omega)\Big(1-\frac{\sum_{l=1}^{k}b_{[l]}+\sum_{l=1}^{n-k-1}b_{[k+l]}}{\sum_{l=1}^{n}b_{[l]}}\Big)^{\alpha_1}n^{\alpha_2}$$
$$+\Big[b_{[n]}-b_{[n]}(1-\omega)\Big(1-\frac{\sum_{l=1}^{k}b_{[l]}+\sum_{l=1}^{n-k-1}b_{[k+l]}}{\sum_{l=1}^{n}b_{[l]}}\Big)^{\alpha_1}n^{\alpha_2}\Big]M_{[n]}$$
$$=A+\sum_{i=1}^{n-k}a_{[k+i]}Z_{[a,k,i-1]}+b_{[n]}\Big[Z_{[b,k,n-k-1]}(1-M_{[n]})+M_{[n]}\Big]$$
$$\geq A+\sum_{i=1}^{n-k}a_{[k+i]}Z_{[a,k,i-1]}+b_{[n]}\Big[Z_{[b,k,n-k-1]}(1-M^{'})+M_{[k]}\Big] \quad (12)$$

where $M^{'}=1-e^{-\sigma(\sum_{l=1}^{n}I_{[l]}+\sum_{l=1}^{n-k}a_{[k+l]})}$. $M^{'}$ and $M_{[k]}$ are functions so that $M_{[n]}\leq M^{'}$ and $M_{[n]}\geq M_{[k]}$. $M^{'}$ contains not only the maximum idle time of the unscheduled sequence but also the sum of the idle times of $\pi^P$. $M_{[k]}$ is only the sum of the idle time of the current partial sequence $\pi^P$.

Because $A$ is known in Equation (12), the lower bound of the makespan of $\pi^P$ only depends on the remaining part. Since the function $Z_{[a,k,i-1]}$ decreases as $i$ increases, $\sum_{i=1}^{n-k}a_{[k+i]}Z_{[a,k,i-1]}$ can be minimized by sequencing the unscheduled jobs on machine 1 using the SPT rule, and $b_{[n]}\Big[Z_{[b,k,n-k-1]}(1-M^{'})+M_{[k]}\Big]$ can be minimized by sequencing the unscheduled jobs on machine 2 using the longest processing time (LPT) rule. Therefore, we obtain the first lower bound $LB_1$:

$$LB_1=A+\sum_{i=1}^{n-k}a_{[k+i]}Z_{[a,k,i-1]}+b_{[n]}\Big[Z_{[b,k,n-k-1]}(1-M^{'})+M_{[k]}\Big]$$

where $a_{(k+1)}\leq a_{(k+2)}\leq\ldots\leq a_{(n)}$ means that the unscheduled jobs on machine 1 are arranged in non-decreasing order of processing times, and $b_{(k+1)}\geq b_{(k+2)}\geq\ldots\geq b_{(n)}$ implies that the unscheduled jobs on machine 2 are arranged in the non-increasing order of processing times.

However, $LB_1$ is not tight.

(i) When the learning effect factor becomes smaller than $\theta$, i.e., $S_{[a,r-1]}<\theta$, we can obtain another lower bound by replacing $S_{[a,r-1]}$ of the above process with $\theta$:

$$LB_2=A+\theta\sum_{i=1}^{n-k}a_{(k+i)}+b_{(n)}[\theta(1-M^{'})+M_{[k]}] \quad (13)$$

(ii) When the forgetting effect factor is negligible, we can obtain the following lower bound:

$$LB_3=A+\sum_{i=1}^{n-k}a_{[k+i]}Z_{[a,k,i-1]}+b_{[n]}Z_{[b,k,n-k-1]}$$

(iii) When there is no forgetting effect and the learning effect factor becomes smaller than $\theta$, we can get the lower bound below:

$$LB_4=A+\theta\sum_{i=1}^{n-k}a_{(k+i)}+b_{(n)}\theta \quad (14)$$

(iv) If no idle time is considered on machine 2 for the unscheduled sequence, we develop the following lower bound:

$$LB_5=B+\sum_{i=1}^{n-k}b_{[k+i]}\Big[Z_{[b,k,i-1]}(1-M_{[k]})+M_{[k]}\Big] \quad (15)$$

(v) When no idle time is considered on machine 2 for the unscheduled sequence and the learning effect factor becomes smaller than $\theta$, we construct another lower bound below:

$$LB_6=B+\sum_{i=1}^{n-k}b_{(k+i)}[(1-M_{[k]})\theta+M_{[k]}] \quad (16)$$

In equation (15), $b_{(k+1)}\leq b_{(k+2)}\leq\ldots\leq b_{(n)}$ illustrates that unscheduled jobs on machine 2 are arranged in non-decreasing order of the processing times. The maximum value from the equations (IV-C1)-(16) is set as the lower bound of $\pi^P$, i.e.,

$$LB=\max\{LB_1,LB_2,LB_3,LB_4,LB_5,LB_6\} \quad (17)$$

*2) Branch and Bound algorithm:* The branch and bound algorithm starts from the best solution obtained after applying the four previous heuristics. A sequence is constructed by the depth-first strategy from the first position. During the search process, a node of the tree is eliminated or expanded based on the lower bound. A complete solution is constructed when the current node is a leaf node, which substitutes the current best solution found so far. The branch and bound algorithm is described in Algorithm 5.

**Algorithm 5:** Branch and Bound Algorithm

**begin**
- Perform JIH, JSH, GIH and GSH to obtain the initial solution;
- Start the assignment of jobs at the beginning of a schedule and move forward one step at a time;
- In the $k$th level node, the first $k$ positions are occupied by $k$ specific jobs. Select one of the remaining $n-k$ jobs for the node at level $k+1$;
- Calculate the lower bound for the node. If the lower bound for an unfathomed partial schedule is larger than the initial solution, eliminate the node. Calculate the objective function value of the completed schedule, and if it is less than the initial solution, replace it as the new solution, otherwise, eliminate it;
- Continue until all nodes have been explored, and the solution that ultimately remains is optimal

## D. An Example

We assume 5 pairs of shoes are sequentially processed on two machines with normal processing times shown in Table I:

TABLE I
PROCESSING TIMES OF SHOES ON TWO MACHINES

| Number | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Machine 1 | 44 | 35 | 30 | 53 | 51 |
| Machine 2 | 31 | 40 | 38 | 44 | 26 |

Actual processing times of a sequence can be obtained according to the learning effect models represented by Equations (9) and (10) with parameters $\theta$=0.75, $\omega$=0.15, $\alpha_1$=1.001, $\alpha_2$=−0.515, $\sigma$=0.02. Jobs' processing times depend on their positions in the sequence. For different schedules constructed by the Johnson rule, Greedy, JIH, JSH, GIH and GSH and Branch & Bound, actual processing times of the jobs are listed in Table II.

TABLE II
ACTUAL PROCESSING TIMES FOR DIFFERENT SCHEDULES

| | | | | | | |
|---|---|---|---|---|---|---|
| Johnson Rule | Schedule | 3 | 2 | 4 | 1 | 5 |
| | Machine 1 | 25.50 | 26.25 | 39.75 | 33.00 | 38.25 |
| | Machine 2 | 32.30 | 30.00 | 33.78 | 23.80 | 21.40 |
| Greedy | Schedule | 3 | 2 | 1 | 5 | 4 |
| | Machine 1 | 25.50 | 26.25 | 33.00 | 38.25 | 39.75 |
| | Machine 2 | 32.30 | 30.00 | 23.25 | 20.88 | 38.06 |
| JIH | Schedule | 3 | 4 | 1 | 2 | 5 |
| | Machine 1 | 25.50 | 39.75 | 33.00 | 38.25 | 26.25 |
| | Machine 2 | 32.30 | 34.52 | 24.32 | 21.63 | 33.87 |
| JSH | Schedule | 3 | 4 | 1 | 2 | 5 |
| | Machine 1 | 25.50 | 39.75 | 33.00 | 38.25 | 26.25 |
| | Machine 2 | 32.30 | 34.52 | 24.32 | 21.63 | 33.87 |
| GIH | Schedule | 4 | 3 | 2 | 1 | 5 |
| | Machine 1 | 45.05 | 22.50 | 26.25 | 38.25 | 33.00 |
| | Machine 2 | 37.40 | 28.50 | 30.00 | 19.50 | 23.93 |
| GSH | Schedule | 4 | 2 | 1 | 3 | 5 |
| | Machine 1 | 45.05 | 26.25 | 33.00 | 38.25 | 22.50 |
| | Machine 2 | 37.40 | 30.00 | 23.25 | 20.33 | 30.07 |
| Branch&Bound | Schedule | 3 | 2 | 1 | 4 | 5 |
| | Machine 1 | 25.50 | 26.25 | 33.00 | 39.59 | 38.25 |
| | Machine 2 | 32.30 | 30.00 | 23.25 | 35.59 | 21.29 |

Makespan of the schedule (3,2,4,1,5) obtained by the Johnson rule is 244 and that of the sequence (3,2,1,4,5) generated by the Branch & Bound is 183.88.

## V. COMPUTATIONAL EXPERIMENTS

In this section, the proposed methods are evaluated with randomly generated instances. We compare the branch and bound algorithm (BB) with the four heuristics on small instances because it would be far too time-consuming as $n$ increases, e.g., the maximum computation time of BB could be more than 1700s for $n$=12. Furthermore, the four heuristics are compared over large size problems. All the involved algorithms are coded in Java and run on Intel(R) Core(TM) i5-3470 CPU @ 3.20GHz with 1GB RAM on Windows 7. The job processing times on the machines are randomly generated from a uniform distribution over [1,100] as it is common in the scheduling literature.

### A. Performance comparison on small-scale problems

The proposed BB method obtains the optimum solution for each instance. BB is compared with the four presented heuristics (JIH, JSH, GIH and GSH), the Johnson rule and Greedy on small instances with the number of jobs $n$ taking values from $\{8,9,10,11,12\}$. However, because the developed model is non-linear, the proposed BB is not compared with CPLEX or with any other black-box mathematical programming solver. Twenty replications are randomly generated for each $n$. The parameters of the learning-forgetting model with experience are set as follows: the experience factor $\omega\in\{0.1,0.15,0.2\}$, the learning threshold $\theta\in\{0.25,0.5,0.75\}$ [19], the first learning index $\alpha_1\in\{1.001,1.01,1.1\}$ [19], the second learning index $\alpha_2\in\{-0.152,-0.322,-0.515\}$ [33] and the forgetting index $\sigma\in\{0.01,0.15,0.2\}$, i.e., there are $3^5$=243 parameter combinations. Therefore, there are $20\times5$=100 instances tested with the $3^5$=243 index combinations and for all 7 tested algorithms resulting in a set of data with 170100 results.

Since BB gets the optimum solution $V_i^*$ for instance $i$, we just show the number of extended nodes in the search tree and the computation times (in seconds) needed, i.e., Mean Time, Max Time, Mean Nodes and Max times. The solution of instance $i$ obtained by heuristic $H$ is denoted as $V_i(H)$. The relative percentage deviation (RPD), commonly used for effectiveness evaluation for scheduling algorithms, is adopted.

$$RPD=\frac{V_i(H)-V_i^*}{V_i^*}\times100\% \tag{18}$$

The results of the seven algorithms are illustrated in Table III, in which ARPD is the average RPD of the instances for each size $n$. The average computation times of the heuristic algorithms are very short for the small instances. They are so small that they cannot be reliably measured so they are not given in Table III.

It can be observed from Table III that the proposed branch and bound algorithm solves a problem with up to 12 jobs in an acceptable amount of time. However, the execution time (both the mean and the maximum) and the number of nodes (both the mean and the maximum) increase dramatically as the number of jobs increases. Similarly, the mean execution time and the mean number of nodes increase dramatically as $\theta$ increases. The other four parameters ($\alpha_1$, $\alpha_2$, $\omega$, $\sigma$) exert little influence on the four indices of the BB.

Among the six heuristics, JIH and JSH have similar ARPDs for each parameter case and they outperform the other four (the best result is given in bold). The ARPDs are almost all lower than 1% except for the $\theta$=0.25 case. Though the ARPD of Johnson is 0.16% when $\theta$=0.75, those of JIH and JSH are only 0.01% and 0.02% in that case. The Greedy is the worst among the compared methods. Though GSH is slightly better than GIH and Johnson, they show similar performance in ARPD. Johnson, JIH and JSH outperform Greedy, GIH and GSH respectively, on small-sized problems. The ARPD of each heuristic decreases with the increase in learning threshold $\theta$. Johnson and Greedy Rules decrease the most. The results also indicate that the learning threshold $\theta$ has a great influence on the efficiency of the heuristics. The ARPDs of the six heuristics show no significant difference in the experience factor $\omega$, the learning index $\alpha_1$, the learning index $\alpha_2$ and the forgetting index $\sigma$. In other words, these parameters have little effect on the performance of heuristics.

We compare the algorithms further by the Analysis of Variance (ANOVA) technique, which is a very robust parametric procedure. There are a number of hypotheses that should ideally be met by the experimental data. Among these, the main three are (in order of importance): independence of the residuals, homoscesdasticity or homogenity of the factor's levels variance and normality in the residuals of the model. Apart from a slight non-normality in the residuals, we can accept all hypotheses easily. The response variable in the experiment is the RPD for each algorithm in each instance.

TABLE III
PERFORMANCE COMPARISONS ON SMALL INSTANCES.

| Param | Values | Branch & Bound | | | | ARPD | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Mean Time | Max time | Mean Nodes | Max Nodes | Greedy | GIH | GSH | Johnson | JIH | JSH |
| $n$ | 8 | 0.02 | 0.17 | 1826 | 13714 | 10.47 | 3.08 | 2.37 | 2.76 | **0.33** | 0.53 |
| | 9 | 0.13 | 1.75 | 10807 | 107003 | 11.89 | 3.80 | 2.58 | 3.38 | **0.37** | 0.40 |
| | 10 | 0.74 | 10.49 | 57920 | 758082 | 8.37 | 4.20 | 3.12 | 2.33 | 0.66 | **0.64** |
| | 11 | 14.43 | 143.95 | 1019567 | 9757849 | 6.69 | 1.93 | 1.32 | 2.77 | 0.38 | **0.30** |
| | 12 | 129.55 | 1780.59 | 7961833 | 108070665 | 6.55 | 2.32 | 1.73 | 2.53 | 0.47 | **0.29** |
| $\theta$ | 0.25 | 0.25 | 17.59 | 17496 | 1246404 | 11.47 | 2.91 | 2.21 | 6.61 | 1.13 | **1.02** |
| | 0.5 | 4.01 | 1743.67 | 260095 | 108070665 | 8.34 | 3.43 | 2.38 | 1.50 | **0.19** | 0.26 |
| | 0.75 | 82.67 | 1780.59 | 5153581 | 104476074 | 6.57 | 2.86 | 2.07 | 0.16 | **0.01** | 0.02 |
| $\omega$ | 0.1 | 27.90 | 1780.59 | 1731177 | 104476074 | 8.81 | 3.03 | 2.13 | 2.99 | 0.48 | **0.46** |
| | 0.15 | 29.46 | 1722.22 | 1844283 | 101153008 | 8.79 | 3.07 | 2.23 | 2.75 | 0.45 | **0.44** |
| | 0.2 | 29.56 | 1743.67 | 1855711 | 108070665 | 8.78 | 3.10 | 2.30 | 2.52 | **0.40** | **0.40** |
| $\alpha_1$ | 1.001 | 29.45 | 1779.13 | 1803031 | 104476074 | 8.79 | 3.06 | 2.22 | 2.77 | 0.45 | **0.43** |
| | 1.01 | 28.64 | 1699.35 | 1797965 | 104476074 | 8.79 | 3.06 | 2.22 | 2.77 | 0.45 | **0.43** |
| | 1.1 | 28.83 | 1780.59 | 1830175 | 108070665 | 8.79 | 3.07 | 2.23 | 2.73 | **0.43** | **0.43** |
| $\alpha_2$ | -0.515 | 32.47 | 1780.59 | 1572148 | 102478930 | 8.91 | 3.22 | 2.27 | 3.15 | **0.49** | 0.51 |
| | -0.322 | 29.64 | 1778.95 | 1839400 | 104476074 | 8.85 | 3.14 | 2.27 | 2.91 | 0.50 | **0.45** |
| | -0.152 | 24.80 | 1710.69 | 2019623 | 108070665 | 8.62 | 2.83 | 2.12 | 2.20 | **0.33** | 0.34 |
| $\sigma$ | 0.01 | 29.07 | 1780.19 | 1817436 | 108070665 | 8.15 | 2.88 | 1.99 | 2.58 | 0.42 | **0.40** |
| | 0.015 | 29.07 | 1780.59 | 1815791 | 108070665 | 8.83 | 3.11 | 2.25 | 2.75 | **0.44** | **0.44** |
| | 0.02 | 28.78 | 1778.95 | 1797944 | 108070665 | 9.40 | 3.21 | 2.43 | 2.94 | **0.46** | **0.46** |

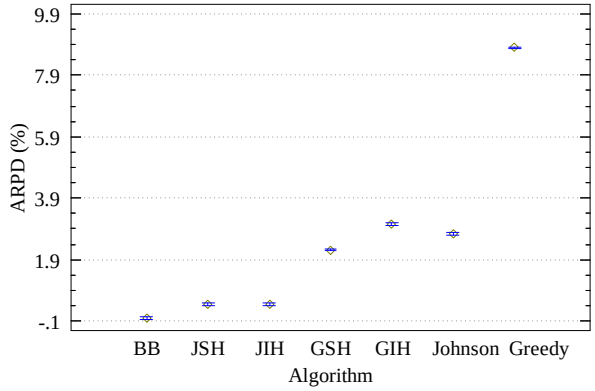


Fig. 2. Means plot of the ARPD and 95% confidence level Tukey's HSD intervals for the compared algorithms on small-scale problems.

All studied factors except $\alpha_1$ in the ANOVA resulted as statistically significant with p-values very close to zero.

The means plot with 95% confidence level Tukey's Honest Significance Differences (HSD) intervals for the ARPD of the compared algorithms is shown in Figure 2. Recall that overlapping intervals indicate that the observed differences in the overlapped means are not statistically significant. Figure 2 summarizes the conclusions indicated in Table III and confirms that the observed differences are statistically significant at the indicated confidence level. The only similarities are between algorithms JSH and JIH which are statistically equivalent. There are no noteworthy interactions between all the studied factors ($n$, $\theta$, $\omega$, $\alpha_1$, $\alpha_2$ and $\sigma$) and the algorithm.

### *B. Performance comparisons of the heuristics on large-scale problems*

To further evaluate the proposed four heuristics, they are compared with the Johnson rule and the Greedy on large-scale problems. Similar to the small-scale case, five different job sizes, $n$=20,50,100,150,200,300 and 400, are tested and 20 replications are randomly generated for each size. The parameter values of the learning-forgetting model with experience are identical to the small-scale case so there are $243\times7\times20\times6$=204120 results in total for all the six compared heuristics. The RPD is adopted for effectiveness evaluation except that now $V_i^*$ is the best solution among the six heuristics for instance $i$ instead of the optimum solution. The results are shown in Table IV.

Table IV illustrates that JSH outperforms the other heuristics in ARPD for all instances and parameter combinations. The ARPD of JSH is no more than 0.06% which basically indicates that in almost all cases it returns the best solution. Similar to the small-scale problems, the performance of the Greedy based heuristics is not better than that of the corresponding Johnson's Rule based ones when $n$<300. ARPDs of JSH and GSH are always lower than those of JIH and GIH respectively, which demonstrates that the swap policy is more effective than the insertion policy for the considered problem. This is a considerable departure from most flow shop problems where, in particular, the insertion neighborhood has been proven to be more effective than the swap or interchange.

As regards the CPU times employed by the algorithms the most important factor is the number of jobs $n$. As we can see, all four presented heuristics use a bit less than 64 seconds on average for the largest instances of 400 jobs. This is expected as all of them have the same computational complexity. The Johnson and Greedy algorithms are very fast, needing a CPU time that is below the measurement threshold. Overall, the presented heuristics can be deemed as very fast.

The six heuristics are also compared using ANOVA. The means plot of the ARPD and 95% confidence level Tukey's HSD intervals for the compared algorithms are shown in

TABLE IV
PERFORMANCE COMPARISONS ON LARGE-SCALE INSTANCES.

| Parameter | Values | Greedy | | Johnson | | GIH | | GSH | | JIH | | JSH | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Time | ARPD | Time | ARPD | Time | ARPD | Time | ARPD | Time | ARPD | Time | ARPD |
| $n$ | 20 | 0.00 | 5.17 | 0.00 | 2.30 | 0.00 | 2.04 | 0.00 | 1.40 | 0.00 | 0.41 | 0.00 | **0.06** |
| | 50 | 0.00 | 3.00 | 0.00 | 1.83 | 0.08 | 1.07 | 0.07 | 0.61 | 0.07 | 0.45 | 0.07 | **0.05** |
| | 100 | 0.00 | 1.90 | 0.00 | 1.55 | 0.76 | 0.68 | 0.75 | 0.36 | 0.75 | 0.31 | 0.75 | **0.03** |
| | 150 | 0.00 | 1.95 | 0.00 | 1.63 | 3.04 | 0.55 | 3.02 | 0.25 | 3.03 | 0.30 | 3.03 | **0.01** |
| | 200 | 0.00 | 1.61 | 0.00 | 1.46 | 7.95 | 0.54 | 7.95 | 0.25 | 7.93 | 0.26 | 7.96 | **0.01** |
| | 300 | 0.00 | 0.02 | 0.00 | 0.03 | 22.35 | 0.01 | 22.11 | **0.00** | 22.09 | 0.01 | 22.20 | **0.00** |
| | 400 | 0.00 | 0.02 | 0.00 | 0.03 | 63.28 | **0.00** | 62.96 | **0.00** | 62.69 | 0.01 | 62.50 | **0.00** |
| $\theta$ | 0.25 | 0.00 | 3.44 | 0.00 | 3.19 | 18.38 | 0.97 | 18.41 | 0.47 | 18.30 | 0.66 | 18.25 | **0.06** |
| | 0.5 | 0.00 | 1.52 | 0.00 | 0.54 | 18.17 | 0.70 | 18.24 | 0.47 | 18.11 | 0.08 | 18.06 | **0.01** |
| | 0.75 | 0.00 | 0.88 | 0.00 | 0.04 | 17.84 | 0.42 | 17.81 | 0.29 | 17.72 | **0.00** | 17.61 | **0.00** |
| $\omega$ | 0.1 | 0.00 | 2.03 | 0.00 | 1.42 | 10.93 | 0.72 | 10.98 | 0.41 | 10.86 | 0.28 | 10.84 | **0.03** |
| | 0.15 | 0.00 | 1.95 | 0.00 | 1.25 | 21.48 | 0.69 | 21.41 | 0.42 | 21.34 | 0.25 | 21.24 | **0.02** |
| | 0.2 | 0.00 | 1.87 | 0.00 | 1.10 | 21.98 | 0.68 | 22.08 | 0.41 | 21.94 | 0.22 | 21.83 | **0.02** |
| $\alpha_1$ | 1.001 | 0.00 | 1.95 | 0.00 | 1.27 | 17.69 | 0.70 | 17.74 | 0.41 | 17.57 | 0.25 | 17.53 | **0.02** |
| | 1.01 | 0.00 | 1.95 | 0.00 | 1.27 | 18.49 | 0.70 | 18.49 | 0.41 | 18.43 | 0.25 | 18.31 | **0.02** |
| | 1.1 | 0.00 | 1.94 | 0.00 | 1.24 | 18.21 | 0.69 | 18.23 | 0.41 | 18.15 | 0.24 | 18.07 | **0.02** |
| $\alpha_2$ | -0.515 | 0.00 | 1.75 | 0.00 | 0.97 | 18.07 | 0.67 | 18.09 | 0.43 | 17.97 | 0.16 | 17.92 | **0.01** |
| | -0.322 | 0.00 | 2.03 | 0.00 | 1.40 | 18.29 | 0.72 | 18.31 | 0.42 | 18.20 | 0.27 | 18.14 | **0.02** |
| | -0.152 | 0.00 | 2.07 | 0.00 | 1.40 | 18.03 | 0.71 | 18.07 | 0.39 | 17.97 | 0.32 | 17.85 | **0.03** |
| $\sigma$ | 0.01 | 0.00 | 1.83 | 0.00 | 1.25 | 18.22 | 0.67 | 18.25 | 0.35 | 18.17 | 0.25 | 18.11 | **0.02** |
| | 0.015 | 0.00 | 1.96 | 0.00 | 1.26 | 18.36 | 0.70 | 18.41 | 0.43 | 18.24 | 0.25 | 18.24 | **0.02** |
| | 0.02 | 0.00 | 2.06 | 0.00 | 1.26 | 17.81 | 0.72 | 17.80 | 0.45 | 17.73 | 0.25 | 17.56 | **0.02** |

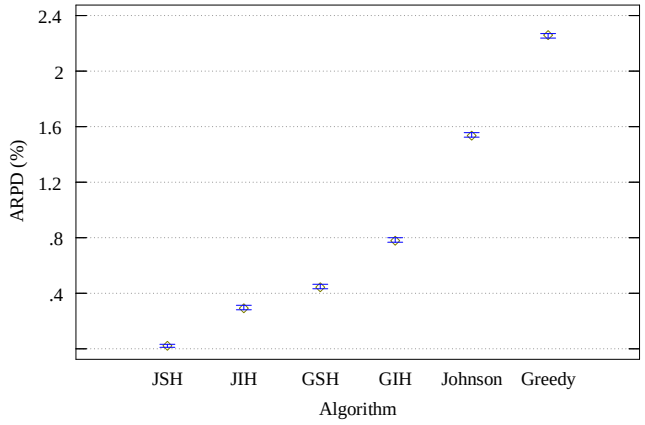


Fig. 3. Means plot of the ARPD and 95% confidence level Tukey's HSD intervals for the compared algorithms on large-scale problems.

Figure 3.

Figure 3 implies that the observed differences in ARPDs of the compared algorithms of Table IV are indeed significantly different. JSH is clearly the best performer and given that its running time is very similar to the next three other best methods it is the preferred heuristic. Johnson is an alternative to Greedy given its better performance and and it can also be considered as an option over JSH when extremely fast CPU times are required.

Lastly, we analyze the effect of the studied instance characteristics and parameters of the learning-forgetting model with experience in the ANOVA. Most interactions between the algorithms and these factors are statistically significant. They are depicted in Figure 4.

The results show that $n$ and the learning threshold $\theta$ have a significant influence on the efficiency of the heuristic algorithms. However, the interactions are rather weak for the other factors, i.e., the performance of the algorithms is not affected by the values of experience $\omega$, learning index $\alpha_1$, learning index $\alpha_2$ and forgetting index $\sigma$. Therefore, the proposed heuristics are robust as regards these parameters.

## VI. CONCLUSIONS AND FUTURE RESEARCH

We have presented a general learning effect model where the actual job processing time is not only sum-of-processing-time-based and job-position-based, but also depends on a workers' experience. We have shown that the SPT rule provides the optimal sequences for the single-machine makespan and total completion time objectives. We have also proved that the WSPT rule provides the optimal sequence for the total weighted completion time and the EDD rule provides the optimal sequences for the maximum lateness and total tardiness problems under certain agreeable conditions. We have also studied two-machine permutation flow shop scheduling problems with a learning-forgetting effects model where the forgetting effect is caused by the idle time of the second machine. Six lower bounds have been derived and used in a branch-and-bound algorithm to find optimal solutions for small-scale problems of up to 12 jobs. Four heuristics, JIH, JSH, GIH and GSH are proposed to find approximate solutions. Computational results show that JSH outperforms Johnson, Greedy, JIH, GIH and GSH on both small-scale and large-scale problems. The performance of the heuristics is not significantly affected by the parameters of the learning-forgetting model in most cases. The swap policy is better than the insertion policy for the considered problem.

For future research, flow shops of more than two machines ($m$>2) with the learning-forgetting effect and experience is

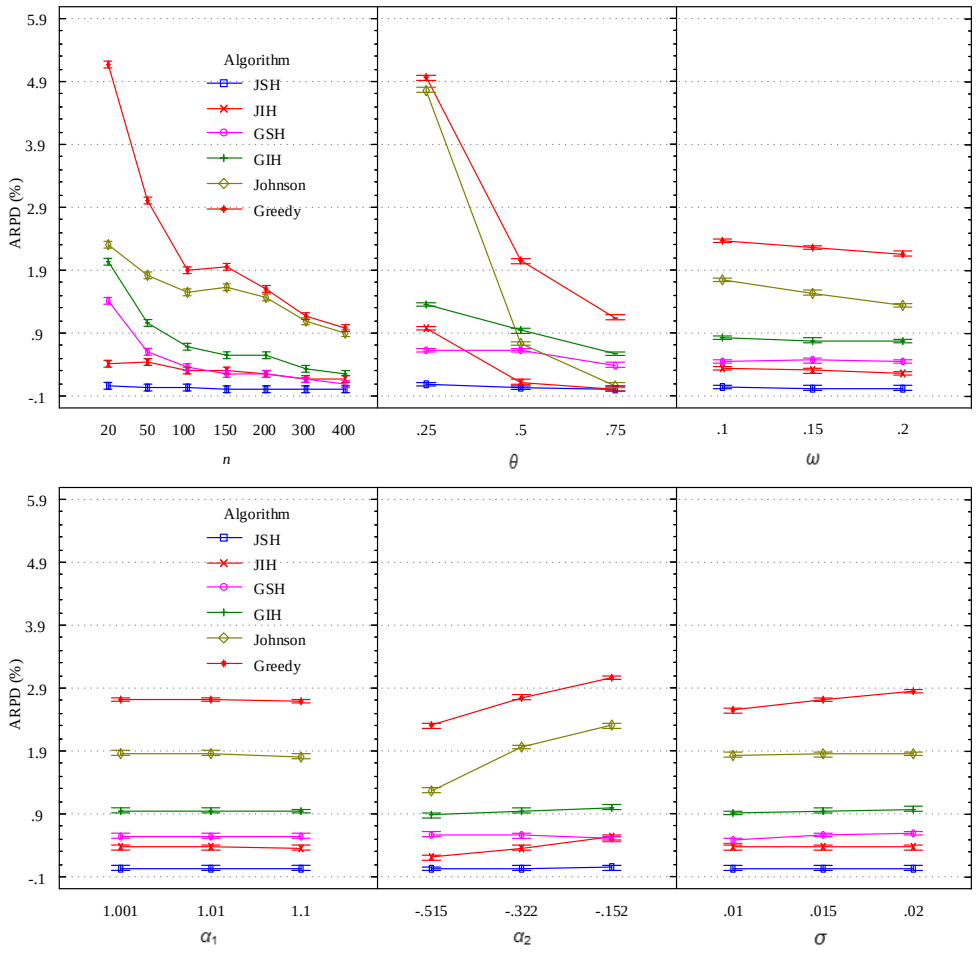


Fig. 4. Interactions between the parameters of the learning-forgetting model with experience on ARPDs with 95% confidence level Tukey's HSD intervals.

the next logical step. Similarly, more general and practical scheduling models are desirable areas of study for future work. Other ideas to consider is even more generalized learning-forgetting effects models where apart from experience one could consider job complexity in the sense that some jobs are very complex and learning takes a longer time and similarly forgetting takes shorter.

## Acknowledgment

This work is supported by the National Natural Science Foundation of China (Grants 61572127, 61272377), the Key Research & Development program in Jiangsu Province (No. BE2015728) and Collaborative Innovation Center of Wireless Communications Technology. Rubén Ruiz is supported by the Spanish Ministry of Economy and Competitiveness, under the project "RESULT - Realistic Extended Scheduling Using Light Techniques "(No. DPI2012-36243-C02-01) financed with FEDER funds.

## References

[1] A. B. Badiru, "Computational survey of univariate and multivariate learning curve models," *Engineering Management, IEEE Transactions on*, vol. 39, no. 2, pp. 176–188, 1992.

[2] D. Biskup, "A state-of-the-art review on scheduling with learning effects," *European Journal of Operational Research*, vol. 188, no. 2, pp. 315–329, 2008.

[3] ——, "Single-machine scheduling with learning considerations," *European Journal of Operational Research*, vol. 115, no. 1, pp. 173–178, 1999.

[4] G. Mosheiov, "Scheduling problems with a learning effect," *European Journal of Operational Research*, vol. 132, no. 3, pp. 687–693, 2001.

[5] C. Zhao, Q. Zhang, and H. Tang, "Machine scheduling problems with learning effects," *Dynamics of Continuous, Discrete and Impulsive Systems, Series A: Mathematical Analysis*, vol. 11, no. 5-6, pp. 741–750, 2004.

[6] W.-H. Kuo and D.-L. Yang, "Minimizing the total completion time in a single-machine scheduling problem with a time-dependent learning effect," *European Journal of Operational Research*, vol. 174, no. 2, pp. 1184–1190, 2006.

[7] A. Bachman and A. Janiak, "Scheduling jobs with position-dependent processing times," *Journal of the Operational Research Society*, vol. 55, no. 3, pp. 257–264, 2004.

[8] C. Koulamas and G. J. Kyparisis, "Single-machine and two-machine flowshop scheduling with general learning functions," *European Journal of Operational Research*, vol. 178, no. 2, pp. 402–407, 2007.

[9] C.-C. Wu and W.-C. Lee, "Single-machine and flowshop scheduling with a general learning effect model," *Computers & Industrial Engineering*, vol. 56, no. 4, pp. 1553–1558, 2009.

[10] T. Cheng, C.-C. Wu, and W.-C. Lee, "Some scheduling problems with sum-of-processing-times-based and job-position-based learning effects," *Information Sciences*, vol. 178, no. 11, pp. 2476–2487, 2008.

[11] Y. Yin, D. Xu, K. Sun, and H. Li, "Some scheduling problems with general position-dependent and time-dependent learning effects," *Information Sciences*, vol. 179, no. 14, pp. 2416–2425, 2009.

[12] X. Zhang and G. Yan, "Machine scheduling problems with a general learning effect," *Mathematical and Computer Modelling*, vol. 51, no. 1, pp. 84–90, 2010.

[13] T. E. Cheng and G. Wang, "Single machine scheduling with learning

effect considerations," *Annals of Operations Research*, vol. 98, no. 1-4, pp. 273–290, 2000.

[14] A. Janiak and R. Rudek, "Experience-based approach to scheduling problems with the learning effect," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 39, no. 2, pp. 344–357, 2009.

[15] ——, "A note on a makespan minimization problem with a multi-ability learning effect," *Omega*, vol. 38, no. 3, pp. 213–217, 2010.

[16] J.-B. Wang, D. Wang, L.-Y. Wang, L. Lin, N. Yin, and W.-W. Wang, "Single machine scheduling with exponential time-dependent learning effect and past-sequence-dependent setup times," *Computers & Mathematics with Applications*, vol. 57, no. 1, pp. 9–16, 2009.

[17] W.-C. Lee, C.-C. Wu, and P.-H. Hsu, "A single-machine learning effect scheduling problem with release times," *Omega*, vol. 38, no. 1, pp. 3–11, 2010.

[18] D.-L. Yang, T. Cheng, and W.-H. Kuo, "Scheduling with a general learning effect," *The International Journal of Advanced Manufacturing Technology*, vol. 67, no. 1-4, pp. 217–229, 2013.

[19] C.-C. Wu, W.-H. Wu, P.-H. Hsu, and K. Lai, "A two-machine flowshop scheduling problem with a truncated sum of processing-times-based learning function," *Applied Mathematical Modelling*, vol. 36, no. 10, pp. 5001–5014, 2012.

[20] T. Cheng, C.-C. Wu, J.-C. Chen, W.-H. Wu, and S.-R. Cheng, "Two-machine flowshop scheduling with a truncated learning function to minimize the makespan," *International Journal of Production Economics*, vol. 141, no. 1, pp. 79–86, 2013.

[21] P.-J. Lai and W.-C. Lee, "Single-machine scheduling with general sum-of-processing-time-based and position-based learning effects," *Omega*, vol. 39, no. 5, pp. 467–471, 2011.

[22] J.-B. Wang and J.-J. Wang, "Scheduling jobs with a general learning effect model," *Applied Mathematical Modelling*, vol. 37, no. 4, pp. 2364–2373, 2013.

[23] P.-J. Lai and W.-C. Lee, "Single-machine scheduling with learning and forgetting effects," *Applied Mathematical Modelling*, vol. 37, no. 4, pp. 4509–4516, 2012.

[24] W.-H. Yang and S. Chand, "Learning and forgetting effects on a group scheduling problem," *European Journal of Operational Research*, vol. 187, no. 3, pp. 1033–1044, 2008.

[25] L. B. Resnick, "Mathematics and science learning: A new conception," *Science*, vol. 220, no. 4596, pp. 477–478, 1983.

[26] N. Neuenhaus, C. Artelt, and W. Schneider, "The impact of cross-curricular competences and prior knowledge on learning outcome." *International Journal of Higher Education*, vol. 2, no. 4, pp. 214–227, 2013.

[27] W. Schneider, J. Körkel, and F. E. Weinert, "Domain-specific knowledge and memory performance: A comparison of high-and low-aptitude children." *Journal of Educational Psychology*, vol. 81, no. 3, pp. 306–312, 1989.

[28] J.-B. Wang and J.-J. Wang, "Scheduling jobs with a general learning effect model," *Applied Mathematical Modelling*, vol. 37, no. 4, pp. 2364–2373, 2013.

[29] T. Cheng, S.-R. Cheng, W.-H. Wu, P.-H. Hsu, and C.-C. Wu, "A two-agent single-machine scheduling problem with truncated sum-of-processing-times-based learning considerations," *Computers and Industrial Engineering*, vol. 60, no. 4, pp. 534–541, 2011.

[30] C. D. Bailey, "Forgetting and the learning curve: A laboratory study," *Management Science*, vol. 35, no. 3, pp. 340–352, 1989.

[31] A. Shtub, N. Levin, and S. Globerson, "Learning and forgetting industrial skills: an experimental model," *International Journal of Human Factors in Manufacturing*, vol. 3, no. 3, pp. 293–305, 1993.

[32] S. M. Johnson, "Optimal two-and three-stage production schedules with setup times included," *Naval research logistics quarterly*, vol. 1, no. 1, pp. 61–68, 1954.

[33] J.-B. Wang, P. Ji, T. Cheng, and D. Wang, "Minimizing makespan in a two-machine flow shop with effects of deterioration and learning," *Optimization Letters*, vol. 6, no. 7, pp. 1393–1409, 2012.