Proceedings of the 2017 Winter Simulation Conference
W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, eds.

THE VALUE OF REAL-TIME DATA IN STOCHASTIC FLOWSHOP SCHEDULING: A SIMULATION STUDY FOR MAKESPAN

Jose M Framinan
Paz Perez-Gonzalez
Victor Fernandez-Viagas Escudero

Industrial Management, School of Engineering
University of Sevilla
Ave. Descubrimientos s/n
Seville, SPAIN

ABSTRACT

This paper presents an effort to assess how real-time data can be used to re-sequence jobs in a flowshop where processing times are stochastic and the objective is the minimisation of the makespan. By conducting extensive simulation experiments, we try to quantify the advantages of collecting real-time data on the actual completion times of jobs in the shop in order to re-sequence the jobs remaining to be processed. The results show that the benefit of re-sequencing is greatly influenced by the variability of the processing times on the shop floor: There is little advantage when the variability is very high but, for low and intermediate variability levels, re-sequencing provides a way to improve the performance of the initial solution. These results may serve to establish limits on the advantages of using real-time data for improving initial sequencing decisions, at least within the boundaries of our experiments.

1 INTRODUCTION

Recent developments in industrial informatics and Information and Communications Technology – such as Industry 4.0 – seem to anticipate a situation in which the shop floor status in many manufacturing companies could be instantly available (Chen et al. 2016). The potential challenges and advantages of the use of this real-time data in production management have been discussed in different contributions, such as e.g. Waschneck et al. (2017). In this paper, however, we focus on an aspect which, to the best of our knowledge, has not been addressed so far, i.e. the limits on the use of such real-time data to cope with the inherent stochastic behaviour of the processing times of many manufacturing processes. More specifically, we wish to assess whether there are advantages to using information about the jobs already processed in some machines to modify the schedule of the jobs yet to be processed. Although, in principle, it seems advantageous to use the new information to modify the existing sequence, such advantages have to be quantified, as on the one hand it may be challenging to perform a real-time, efficient re-sequencing of the jobs, and on the other hand, re-sequencing modifies the initial sequence and induces nervousness on the shop floor. Finally, even if more data is available, re-sequencing still has to rely on estimates regarding the completion times of the jobs not yet processed, causing its theoretical advantages to be diluted in practice.

In this work, we focus on an specific shop floor layout (the permutation flowshop), and on a specific scheduling objective (makespan). In a permutation flowshop schedule, all jobs must follow the same route across all machines, so once a job enters into the shop floor (i.e. in the first machine), then it is processed according to the same sequence in the rest of the machines. Since the sequence of the jobs does not change once they enter in the shop, the point in time where the shop floor information can be used is when a job has completed its processing on the first machine, leaving this machine free for the subsequent jobs. In this situation, we wish to investigate whether it is more beneficial to maintain the initial sequence, or to

perform a re-sequencing of the remaining jobs by taking into account the unavailability of the machines in the shop caused by the processing of the previous jobs. To do so, we carry out a series of simulations for different settings.

The remainder of the paper is as follows: Section 2 presents the background and formalises out problem. In Section 3 we present the procedure to carry out the re-sequencing procedure. The simulation experiments carried out are described in Section 5, while in Section 6 we present the conclusions of our work and point out future research lines.

2 BACKGROUND AND PROBLEM DESCRIPTION

As mentioned in the previous section, we will focus on the permutation flowshop for the makespan objective. The choice of the layout and the objective is driven by their predominance both in academia and in practice, being undoubtedly one of the most researched problems in the OR and MS area (see e.g. Fernandez-Viagas et al. 2017 for a recent review on the topic).

In the classical permutation flowshop scheduling problem, there are n jobs to be processed sequentially on a set of m machines. The permutation constraint indicates that the sequence of the jobs is the same for all machines. Additionally, it is usually assumed that machines are always available, and that there are no set-up times. For a complete list of the assumptions in the classical permutation flowshop problem, see e.g. Dudek and Teuton (1964).

Due to the permutation constraint, a sequence of jobs Π specifies a complete solution to the problem. Let us denote $C_{ij}(\Pi)$ the completion time on machine i of job in order j in sequence Π. Then, the maximum completion time or makespan of the solution Π can be computed using the following recursive expression:

$$C_{ij}(\Pi) = \max\{C_{i-1,j}; C_{i,j-1}\} + p_{ij} \quad \forall i = 1,\ldots,m, j = 1,\ldots,n \tag{1}$$

where p_{ij} denotes the processing time of job in position j ($j = 1,\ldots,n$) on machine i ($i = 1,\ldots,m$). Clearly, $C_{0,j} = 0$ and $C_{i,0} = 0$. Then, $C_{max} = C_{mn}$.

Among all possible sequences, the goal is to find the one yielding the minimum makespan. This decision problem is known to be NP-hard for $m > 2$.

In the stochastic counterpart of the permutation flowshop scheduling problem for the makespan objective, the processing times are random variables with expected value $\tilde{p}_{ij}$. The objective is here to minimise the expected makespan. Clearly, since p_{ij} are random variables, so are C_{ij}, see (1). An estimate of the values of C_{ij} – denoted as $\tilde{C}_{ij}$ – can be computed using the expected value of the processing times, i.e.:

$$\tilde{C}_{ij}(\Pi) = \max\{\tilde{C}_{i-1,j}; \tilde{C}_{i,j-1}\} + \tilde{p}_{ij} \quad \forall i = 1,\ldots,m, j = 1,\ldots,n \tag{2}$$

also $\tilde{C}_{0,j} = 0$ and $\tilde{C}_{i,0} = 0$.

The stochastic flowshop scheduling problem has been much less studied than the classical counterpart, and it is much more complex. Apart from a dominance property by Makino (1965) for the (almost trivial) two-jobs case, no exact solution is available without making specific assumptions regarding the processing times' distribution. More specifically, for an exponential distribution of p_{ij} and $m = 2$, Talwar (1967) proposes an exact procedure – known as Talwar's rule – for the problem that was later proved to be optimal by Cunningham and Dutta (1973).

For other distributions, no polynomial-time procedure has been found. For $m = 2$, three heuristics are proposed by Baker and Trietsch (2011) based both in Talwar's rule and in Johnson's rule (Johnson 1954) for the deterministic flowshop, all of them with similar (near optimal) performance. For the general m machine case, Baker and Altheimer (2012) propose three constructive heuristics based on adaptations of the CDS (Campbell et al. 1970) and NEH (Nawaz et al. 1983) heuristics to this problem. Although the computational experiments carried out by the authors show that these heuristics have a similar and near optimal performance, these findings are questioned by Framinan and Perez-Gonzalez (2015), who find that, in many cases, the confidence interval of the results was too wide to safely support these statements.

Furthermore, in their experimentation, Framinan and Perez-Gonzalez (2015) found that the performance of deterministic scheduling procedures do not differ very much to that of specific stochastic procedures, and that these differences decrease as the processing times become more variable (in terms of increasing their coefficient of variation). A result in line with these findings is stated in Juan et al. (2014), who develop a high-performing simheuristic under the assumption that high-quality solutions for the deterministic version of the problem are likely to be high-quality solutions for the stochastic version. A conclusion of these two last contributions is that deterministic procedures for flowshop scheduling for the makespan objective perform consistently well for the stochastic counterpart if the mean processing times are used as input for the deterministic procedure. Finally, Wang et al. (2015) propose a two-stage simulation-based hybrid estimation of distribution algorithm for the problem, although the authors do not compare the performance of their proposal with that of deterministic procedures.

In view of this excellent performance of the deterministic procedures in the stochastic setting, another issue of interest is to study whether these deterministic procedures can be combined with real-time data at hand so their performance can be increased. More specifically, whenever job in position j in sequence Π has completed its processing on the first machine of the flowshop, this machine is available for the next job. In such situations, two options are possible:

- To proceed with the next job in Π, that it, to enter job in position $j+1$ in the first machine. We will refer to this option as to *no re-sequencing*, as it results in sequencing the jobs according to the initial sequence Π.
- To re-sequence the jobs in positions $j+1$ to n in Π to take into account the possible early/tardy completion of the preceding jobs with respect to the initial sequence. Clearly, re-sequencing the remaining jobs is equivalent to solving a decision problem with machine availability constraints. In the first machine, these availability is caused by the actual completion time of job in position j, whereas for the remaining machines $2,\ldots,m$ is an *estimated* availability based on the expected completion time of job in position j in this machine, even though it incorporates some information regarding the actual completion times of the jobs. We refer to this option as to *re-sequencing*.

There are potential merits in both options: On the one hand, re-sequencing uses the available (real-time) data to cope with the uncertainty of the shop floor. On the other hand, deterministic scheduling based on average processing times performs extremely well for the stochastic flowshop, as shown by Framinan and Perez-Gonzalez (2015). Furthermore, re-sequencing may introduce excessive nervousness in the scheduling process with perhaps little practical advantage, given the fact that, although using some additional data, the computation of the completion times of the remaining jobs is still based on estimates.

There are several contributions dealing with the use of real-time information for scheduling decisions. Ovacik and Uzsoy (1994) study the advantages of using dispatching rules that incorporate real-time shop floor information. Cowling and Johansson (2002) investigate, for a single machine setting, the trade-off between rescheduling (i.e. rebuilding a full scheduling from scratch) and schedule repair (i.e. making localised changes to an existing scheduling) in the presence of real-time information. However, note that our research deviates from the majority of the copious body of literature addressing rescheduling decisions (see e.g. Vieira et al. 2003) at least in two aspects: First, the stochastic nature of processing times is not usually considered a so-called *rescheduling factor* (i.e. the unexpected event that triggers the rescheduling process) in the rescheduling literature, and second, we the re-sequencing procedure is not triggered by any unexpected event, but on a regular basis whenever a new job has to enter into the shop. Although some works also consider the completion time of a job in a machine as the event to trigger the rescheduling (see e.g. Ovacik and Uzsoy 1994, or Church and Uzsoy 1992), we are not aware of a work that analyses the boundaries of the potential advantages of re-sequencing with respect to a static, deterministic sequencing.

A final remark is that, clearly, re-sequencing does not come at zero computational costs, as in practical terms means that a new sequence has to be found in near real-time. However, although extremely important in practice, in this research we will set aside this issue and assume that the time can be 'frozen' so a true

re-optimisation of the partial sequence can be conducted. By doing so, we aim to establish the best case regarding the re-sequencing option, as it may turn out that, under certain conditions, it is not profitable to re-sequence even if we could afford very long decision intervals. Note that some work exist (Upasani et al. 2006) on problem reduction procedures that could be applied to reduce the required computational effort, or on the combination of dispatching rules with global optimisation procedures (Barua et al. 2005).

3 RESEQUENCING USING THE AVAILABLE INFORMATION

As discussed in the previous section, upon the completion of the processing of a job in the first machine of the flowshop, the jobs to be yet processed can be re-sequenced so the objective function (the makespan) is minimised. Since some jobs are already being processed in the shop, this re-sequencing problem has to take into account the availability of the machines $2,\ldots,m$, therefore in order to provide the re-sequencing problem with the required data, the availability of the machines have to be computed. More specifically, let $a_i(j)$ be the availability of machine i after the processing of job in position j in the first machine. This availability can be computed in the following manner:

- For the first machine ($i=1$), $a_1(j)$ is given by the real completion time of job in position j on the first machine, i.e. :

$$a_1(j) = C_{1j}.$$

- For the rest of the machines ($i=2,\ldots,m$), two sub-cases can be distinguished:
  - If, by the time the job in position j is finished on machine 1, machine i has completed the processing of job in position $j-1$ (i.e. if $a_1(j) > C_{i,j-1}$), then the expected availability is given by:

$$a_i(j) = \max\{C_{i,j-1}; a_{i-1}(j)\} + \tilde{p}_{ij} \quad i=2,\ldots,m. \tag{3}$$

    Naturally, $C_{i,0}=0$.

    It is clear that $a_i(j) \geq a_{i-1}(j)$, therefore as in the case $a_1(j) > C_{i,j-1}$ it follows that $a_i(j) > C_{i,j-1}$, and Equation (3) can be re-written as simply:

$$a_i(j) = a_{i-1}(j) + \tilde{p}_{ij} \quad i=2,\ldots,m. \tag{4}$$

  - In contrast, if machine i has not completed the processing of job in position $j-1$, we must estimate the completion time of this job. To compute this estimate, let us denote by k the position of the last scheduled job that is completed in machine i at time given by $a_1(j)$, i.e. $C_{i,k} \leq a_1(j)$. This is the last scheduled job for which we have all realized completion times for machines $1,2,\ldots,i$. Let us define $\tilde{C}_{l,k} = C_{l,k}$ for $l=1,\ldots,i$. For the subsequent jobs in position r ($r=k+1,\ldots,j$), their estimated completion times are:
    - For the first machine, note that all jobs in positions $r=k+1,\ldots,j-1$ have completed their processing (otherwise job j cannot have completed its own processing), therefore:

$$\tilde{C}_{1,r} = C_{1,r} \quad \forall \; r=k+1,\ldots,j. \tag{5}$$

    - For the remaining machines ($l=2,\ldots,i$), the completion times of job in position r have to be estimated if their real completion times fall behind $a_1(j)$. Two cases can be distinguished: if $C_{l,r} \leq a_1(j)$ then job in position r has completed its processing time in machine l, therefore $\tilde{C}_{l,r} = C_{l,r}$. In contrast, if $C_{l,r} > a_1(j)$ then the job in position r has not completed its processing on machine l, but its completion time can be estimated using previous estimates of the completion times, i.e. $\max\{\tilde{C}_{l,r-1}, \tilde{C}_{l-1,r}\} + \tilde{p}_{lr}$. Note, however, that this

latter expression may provide a value lower than $a_1(j)$ which does not not reflect the real situation, as we know that the processing time of this job has not completed at time $a_1(j)$. Therefore, we can assume that at least this value should be greater or equal than $a_1(j)$. Equation (6) summarises these options:

$$\tilde{C}_{l,r} = \begin{cases} C_{l,r} & \text{if } C_{l,r} \leq a_1(j) \\ \max\{a_1(j); \max\{\tilde{C}_{l,r-1}; \tilde{C}_{l-1,r}\} + \tilde{p}_{lr}\} & \text{otherwise.} \end{cases} \quad (6)$$

Therefore,

$$a_i(j) = \tilde{C}_{i,j}.$$

This procedure for computing the $a_i(j)$ values can be integrated into the simulation experiment described in the next section.

4 SIMULATION PROCEDURE

In order to assess the differences in makespan value between re-sequencing and not re-sequencing jobs whenever a new job is to be processed on the first machine of a stochastic flowshop, we use discrete event simulation with an integrated optimisation procedure. More specifically, the starting point of the simulation is given by the execution of a deterministic optimisation procedure which attempts to find the best sequence of all jobs using their average processing times. Once this sequence is obtained, the first job in the sequence enters the first machine and the simulation starts. The list of events is given by the completion of a job in the first machine. Every time an event occurs – a job is completed on the first machine –, an optimisation procedure is executed to find the best sequence of the remaining jobs according to the information available at this simulation time. Then, the job to be processed first according to this simulation procedure enters the first machine and the simulation time is advanced until the next event. This procedure is repeated until all jobs have been completed.

The simulation with the integrated optimisation procedure has been coded using C#. The high-level flowchart is given in Figure 1. The detailed procedure is as follows:

1. A sequence is found to approximately minimize the makespan. This sequence is obtained assuming a deterministic behaviour of the flowshop, i.e. $\tilde{p}_{ij}$ the mean processing times of the jobs on each machine are employed to solve the corresponding deterministic permutation flowshop scheduling problem. To ensure a good solution, the IG (Iterated Greedy) algorithm of Ruiz and Stützle (2007) is employed, with a generous amount of computation time (30 seconds). This algorithm is known to be among the best methods for the problem under consideration, so – particularly for the small instances in the testbed – we can be quite confident that the makespan is optimal or close to the optimal (deterministic) value. The parameter values employed in the IG are the same as in Ruiz and Stützle (2007). Let us denote the so-obtained sequence by $\Pi_d := (\pi_1, \ldots, \pi_n)$
2. The actual processing times of the jobs in the machines p_{ij} are sampled from a lognormal distribution of mean $\tilde{p}_{ij}$ and a standard deviation $\sigma = cv \cdot \tilde{p}_{ij}$, where cv is the coefficient of variation (see details in Section 5.1).
3. We set $\Pi_r := \{\pi_1\}$, and for $j = 1, \ldots, n-1$, perform the following actions:
   (a) The realized completion times C_{ij} of job in position j in Π_r are computed according to Equation (1) for job j, i.e.:

$$C_{ij} = \max\{C_{i-1,j}; C_{i,j-1}\} + p_{ij} \quad i = 1, \ldots, m. \quad (7)$$

   where $C_{0,j} = 0$ and $C_{i,0} = 0$.

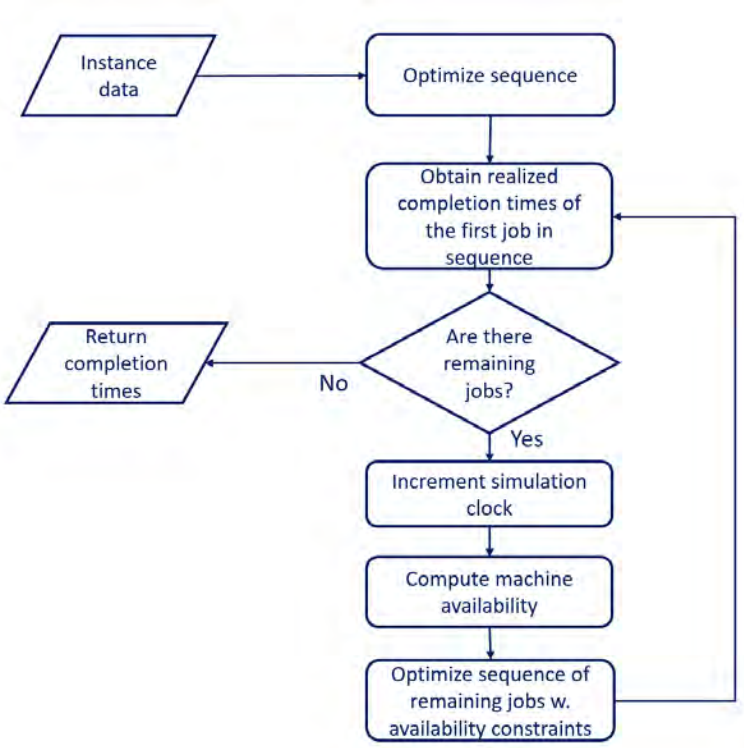

Figure 1: Flowchart of the simulation procedure.

(b) The machine availability $a_i(j)$ after processing job in position j are obtained as described in Section 3.

(c) The problem of minimising the makespan of jobs in positions $j+1,\ldots,n$ in Π_r under the machine availability constraints $a_i(j)$ is solved, and solution $\Omega(j) := (\omega_1,\ldots,\omega_{n-j})$ is obtained. The job w_1 is then appended to Π_r (i.e. $\Pi_r := \Pi_r \cup \{w_1\}$). To solve this problem, we adapt the IG algorithm by changing the objective function to take into account the availability of the machines. More specifically, Equation (2) is modified as follows:

$$\tilde{C}_{ik}(\Pi) = \max\{\tilde{C}_{i-1,k};\tilde{C}_{i,k-1}\} + \tilde{p}_{ik} \quad \forall i = 1,\ldots,m \quad k = j+1,\ldots,n. \tag{8}$$

where $\tilde{C}_{0,k} = 0$ and $\tilde{C}_{i,j} = a_i(j)$.

The so-modified IG algorithm is allowed to run for 30 seconds, as in the initial iteration.

4. The remaining job in $\Omega(n-1)$ is appended to Π_r.
5. Compute both $C_{max}(\Pi_d)$ (no re-sequencing) and $C_{max}(\Pi_r)$ (re-sequencing) to compare the results.

5 COMPUTATIONAL EXPERIENCE

In this section we describe the experiments carried out. In Section 5.1 we discuss the design of the experiments, while in Section 5.2 we present the results of the experiments.

5.1 Design of the Experiments

The procedure described in Section 4 is applied to the testbed of Taillard (1993), that contains 120 flowshop scheduling instances with different numbers of jobs and machines. The processing times in the instances have been chosen so that the instances are difficult to solve in the sense that a lengthy tabu search procedure yields solutions which are relatively far from the lower bound. Out of these 120 instances, the first 60

Table 1: Summary of results: mean and standard deviation of the Average Relative Percentage Deviation (ARPD) for the different problem sizes and coefficient of variations.

		n	20			50			
cv	resch	m	5	10	20	5	10	20	Total
0.5	No-resq	Av	3.03631	2.69330	2.05026	2.85829	2.37480	2.38391	2.56614
		St.Dv	4.15700	3.72463	3.39908	3.66534	3.13044	2.96241	3.53951
	Reseq	Av	1.54765	2.21034	2.03682	1.24106	1.46985	1.38986	1.64926
		St.Dv	3.10714	3.61550	2.98644	2.23291	2.42288	2.40480	2.85458
1	No-resq	Av	4.59810	4.36203	3.50662	3.41475	3.42024	3.46833	3.79501
		St.Dv	7.18744	6.31877	5.69000	5.52147	5.03964	4.57784	5.79781
	Reseq	Av	3.40808	2.91028	3.62562	2.98166	3.04347	2.16599	3.02252
		St.Dv	5.78661	4.79477	5.22471	4.45963	4.59805	3.53344	4.79874
2	No-resq	Av	6.11961	4.86935	4.97551	4.59468	5.38615	4.87318	5.13641
		St.Dv	10.07137	7.29695	7.96069	7.03268	7.82329	7.80416	8.06251
	Reseq	Av	4.20276	4.53836	4.75846	4.81568	4.05375	4.08187	4.40848
		St.Dv	8.12120	7.55553	7.94396	7.44265	7.20930	6.39909	7.46227

ones have been selected for our experimentation. Aside from the enormous computation times required for the simulation of the re-sequencing procedure in the biggest instances (see below), we have opted for this subset of instances in order to ensure that the IG procedure is able to find very good solutions within the allocated CPU time. $n \in \{20,50\}$ and $m \in \{5,10,20\}$ in these instances, with 10 instances of each problem size.

The processing times in the instances are assumed to follow a log normal distribution, which has two parameters: mean μ and standard deviation σ. This distribution has been widely employed to model stochastic processing times (see e.g. Baker and Trietsch 2011, Baker and Altheimer 2012, or Framinan and Perez-Gonzalez 2015). Furthermore, we can easily control the variability of the processing times by setting different levels of coefficient of variation $cv = \frac{\sigma}{\mu}$. More specifically, we set $cv \in \{0.5,1,2\}$, which are assumed to represent a low, medium, a large variability respectively in manufacturing shop floors (see Hopp and Spearman 2008). In this manner, the processing times given in Taillard's tested are interpreted as the mean value of these processing times ($\mu = \tilde{p}_{ij}$).

The computational effort carried out involves solving 30 replications of 60 problem sizes, half of them with 20 jobs, and the other half with 50 jobs. Applying the re-sequencing procedure to each one of the instances requires invoking of the IG algorithm $n-1$ times with a time limit of 30 seconds, which makes a total equivalent computation time of 21.25 days for each coefficient of variation tested.

5.2 Results

The results are summarised in Table 1 in terms of the Average Relative Percentage Deviation (ARPD), where the RPD of instance i for a solution Π is computed as:

$$RPD(\Pi) = \frac{C_{max}(\Pi) - \min\{C_{max}(\Pi_d); C_{max}(\Pi_r)\}}{\min\{C_{max}(\Pi_d); C_{max}(\Pi_r)\}} \cdot 100 \tag{9}$$

where the no re-sequencing and re-sequencing solutions are denoted as Π_d and Π_r respectively. The results are shown with 5 decimal digits according to procedure for reporting point-estimate digits by Song and Schmeiser (2009), as it has been found that low-order digits are meaningless for a threshold of 0.1.

From Table 1 it can be seen that, on average, re-sequencing yields lower ARPD values for all coefficients of variation across all problem sizes. This fact is illustrated in Figure 2, where the 95% confidence intervals

Table 2: Summary of ANOVA.

	Sum of square type III	fd	Quadratic mean	F	Sig.
Corrected model	16405.797[a]	35	468.737	14.238	0.000
Intersection	127034.092	1	127034.092	3858.702	0.000
n	460.278	1	460.278	13.981	0.000
m	154.481	2	77.241	2.346	0.096
cv	12783.888	2	6391.944	194.157	0.000
reseq	1753.010	1	1753.010	53.248	0.000
$n \times m$	17.617	2	8.809	0.268	0.765
$n \times cv$	78.142	2	39.071	1.187	0.305
$n\times$ reseq	2.599	1	2.599	0.079	0.779
$m \times cv$	31.190	4	7.797	0.237	0.918
$m\times$ reseq	130.138	2	65.069	1.976	0.139
$cv\times$ reseq	17.561	2	8.780	0.267	0.766
[a]. Square R= 0.044 (Adjusted square R = 0.041)					

for the aggregated data are shown. These results indicate that the difference between re-sequencing and no re-sequencing is statistically significant.

However, the lowest ARPD values are not consistently lower for all problem sizes, so an Analysis of Variance (ANOVA) is carried out to check which factors influence the results. The main results of the ANOVA are shown in Table 2. The factor that most influences the results is the coefficient of variation, followed by the re-sequencing/no re-sequencing option. The number of jobs has a moderate influence, whereas the number of machines cannot be considered relevant in statistic terms. To illustrate these results, Figure 3 shows that, for low and medium variability ($cv = 0.5$ and $cv = 1$), the difference between re-sequencing and no-resequencing is stastically significant, whereas for a high degree of variability ($cv = 2$), it is not. Still, for $cv = 2$, the ARPD when re-sequencing is around 16% lower than if no re-sequencing is performed.

Regarding the influence of the number of jobs to be scheduled, Figure 4 shows the values and the 95% confidence intervals dissagregated according to the number of jobs. It can be seen that the differences increase with the number of jobs, which speaks for the higher suitability of re-sequencing for scenarios with a higher number of jobs.

Finally, it is worth highlighting that, for almost all problem sizes and coefficient of variations, the standard deviation of the ARPD is lower when re-sequencing is performed than if re-sequencing is not carried out, the only exception being the case where $cv = 2$ and $n = 20$ and $m = 10$, or $cv = 2$ and $n = 50$ and $m = 5$.

6 CONCLUSIONS

This paper presents some exploratory results on the value of using real-time data for conducting re-sequencing decisions, focusing in the case of the permutation flowshop with the objective of minimising the makespan where the processing times are assumed to be stochastic. To do so, we compare the expected performance of a (possibly optimal or near-optimal) sequence obtained assuming a deterministic setting with that of a procedure which uses the real-time data available. This information is captured at the time that a job has completed its processing in the first machine of the flowshop to re-sequence the jobs that have not yet entered the shop. To do so, several simulations including different levels of variability in the processing times, number of jobs and number of machines have been carried out.

Despite the preliminary nature of the research, some conclusions can be highlighted:

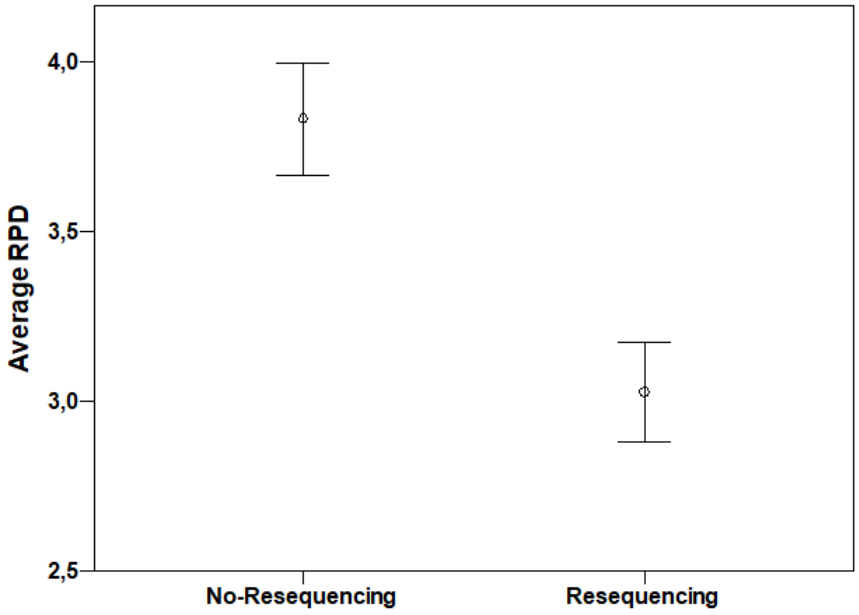

Figure 2: 95% Overall confidence interval plots.

- When variability of the processing times is low to medium (a coefficient of variation between 0.5 and 1 in our research), re-sequencing improves the performance in the shop floor, reducing the ARPD values around 25%. This improvement is statistically significant. When the processing time variability is very high (a coefficient of variation of 2 in our experiments), there are no statistically significant differences between re-sequencing and not re-sequencing, even if the mean and the standard deviation of ARPD are lower if re-sequencing is performed. This good performance of the deterministic procedure may be caused by variance pooling.
- The expected performance of re-sequencing does not seem to be affected by the number of machines in the shop and/or the number of jobs to be scheduled, at least within the limits of our experiments. However, it is affected by the number of jobs, indicating a higher benefit from re-sequencing when the number of jobs is high.

The above conclusions have a number of implications for schedulers and decision makers:

- Investments in gathering real-time data at the scheduling level may pay off depending basically on the variability of the shop floor and, to a lesser extent, on the number of jobs to be scheduled. While there is a range of low/middle variability where this information can be used to provide a more efficient schedule, this is not the case for very high variability. Still, even in the latter case, re-scheduling serves to provide more robust sequences, as the standard deviation of the results is lower.
- Stochastic modelling is sometimes used to capture the fact that there are no accurate estimations of the processing times. In view of the results obtained when the variability is very high, it would be advisable to devote time and resources to provide accurate processing times *before* deploying an (optimisation-based) scheduling system, as the latter may not reap the expected benefits in the case of highly variable processing times.

Finally, a number of possible avenues worth of research include the following:

Figure 3: 95% Confidence interval plots for different coefficient of variations.

- Here we have limited to a specific layout and objective function. It will be interesting to check whether the conclusions obtained also extend to different settings and objectives. Particularly, due date related objectives seem to be very interesting, as the objective function would be a combination of stochastic (processing times) and deterministic (due dates) data, so it may result in a different behaviour than the one obtained for the makespan.
- In the flowshop setting, sometimes is possible to use non-permutation solutions, which would provide more points of decision for re-scheduling (i.e. $n \cdot m - 1$ as compared to $n - 1$ in our experiments) and a more extensive use of available data can be done.
- For the settings with low/medium variability where re-sequencing pays-off, it would be interesting to extend the analysis to the case where the re-sequencing decision can be taken only in a limited number of instants, and/or to develop selective re-sequencing procedures so the re-optimisation is not always carried out every time a job is completed in the first machine.
- Finally, in view of the required running time of the IG – which is unrealistic in many settings –, it would be interesting to analyse the effect of re-sequencing using faster – albeit of poorer performance – heuristics, or whether it is profitable to freeze some jobs in the sequence to give the IG enough running time.

ACKNOWLEDGMENTS

The authors wish to acknowledge the anonymous referees of this communication, as they provided extremely meaningful comments on the earlier version of this document. This research has been funded by the Spanish Ministry of Science and Innovation, under project "ADDRESS" with reference DPI2013-44461-P, and under project "PROMISE" with reference DPI2016-80750-P.

REFERENCES

Baker, K., and D. Altheimer. 2012. "Heuristic Solution Methods for the Stochastic Flow Shop Problem". *European Journal of Operational Research* 216 (1): 172–177.

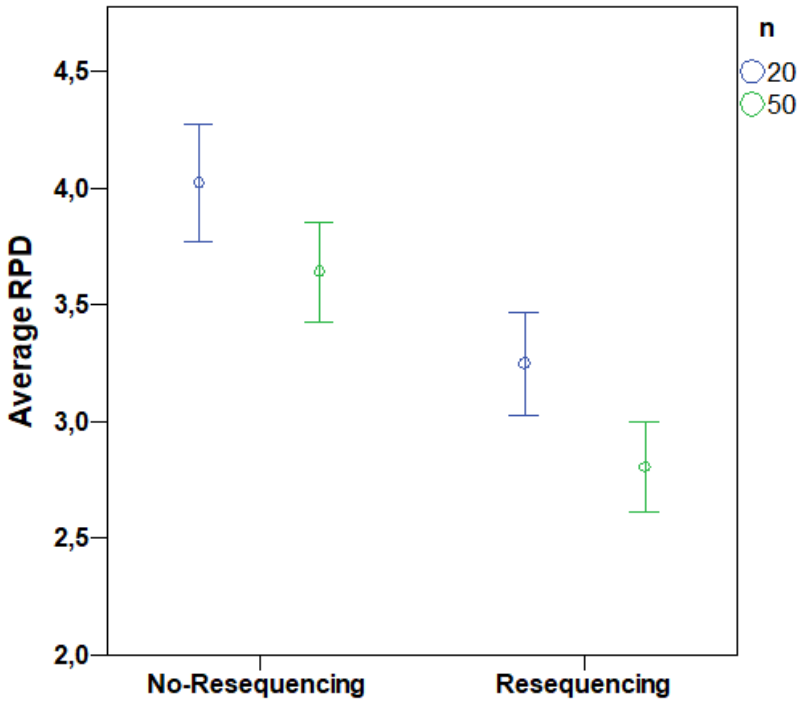

Figure 4: 95% Confidence interval plots for different number of jobs.

Baker, K., and D. Trietsch. 2011. "Three Heuristic Procedures for the Stochastic, Two-machine Flow Shop Problem". *Journal of Scheduling* 14 (5): 445–454.

Barua, A., N. Raghavan, A. Upasani, and R. Uzsoy. 2005. "Implementing Global Factory Schedules in the Face of Stochastic Disruptions". *International Journal of Production Research* 43 (4): 793–818.

Campbell, H., R. Dudek, and M. Smith. 1970. "Heuristic Algorithm for the n Job, m Machine Sequencing Problem". *Management Science* 16 (10): 630–637.

Chen, C.-C., C.-L. Chen, C.-Y. Ciou, and J.-X. Liu. 2016. "Communication Scheduling Scheme based on Big-Data Regression Analysis and Genetic Algorithm for Cyber-Physical Factory Automation". 2016 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2016 - Conference Proceedings, 2603–2608.

Church, L., and R. Uzsoy. 1992. "Analysis of Periodic and Event-Driven Rescheduling Policies in Dynamic Shops". *International Journal of Computer Integrated Manufacturing* 5 (3): 153–163.

Cowling, P., and M. Johansson. 2002. "Using Real Time Information for Effective Dynamic Scheduling". *European Journal of Operational Research* 139 (2): 230–244.

Cunningham, A., and S. Dutta. 1973. "Scheduling Jobs with Exponentially Distributed Processing Times on Two Machines of a Flow Shop". *Naval Research Logistics Quarterly* 16 (1): 69–81.

Dudek, R., and O. Teuton. 1964. "Development of m Stage Decision Rule for Scheduling n Jobs through m Machines". *Operations Research* 12:471.

Fernandez-Viagas, V., R. Ruiz, and J. Framinan. 2017. "A New Vision of Approximate Methods for the Permutation Flowshop to Minimise Makespan: State-of-the-Art and Computational Evaluation". *European Journal of Operational Research* 257 (3): 707–721.

Framinan, J., and P. Perez-Gonzalez. 2015. "On Heuristic Solutions for the Stochastic Flowshop Scheduling Problem". *European Journal of Operational Research* 246 (2): 413–420.

Hopp, W., and M. Spearman. 2008. *Factory Physics (Third Edition)*. Chicago: Irwin.

Johnson, S. 1954. "Optimal Two- and Three-Stage Production Schedules with Setup Times Included". *Naval Research Logistics Quarterly* 1 (1): 61–68.

Juan, A., B. Barrios, E. Vallada, D. Riera, and J. Jorba. 2014. "A Simheuristic Algorithm for Solving the Permutation Flow Shop Problem with Stochastic Processing Times". *Simulation Modelling Practice and Theory* 46:101–117.

Makino, T. 1965. "On a Scheduling Problem". *Journal of the Operations Research Society of Japan* 8:32–44.

Nawaz, M., E. Enscore Jr., and I. Ham. 1983. "A Heuristic Algorithm for the m-Machine, n-Job Flow-Shop Sequencing Problem". *Omega* 11 (1): 91–95.

Ovacik, I., and R. Uzsoy. 1994. "Exploiting Shop Floor Status Information to Schedule Complex Job Shops". *Journal of Manufacturing Systems* 13 (2): 73–84.

Ruiz, R., and T. Stützle. 2007. "A Simple and Effective Iterated Greedy Algorithm for the Permutation Flowshop Scheduling Problem". *European Journal of Operational Research* 177 (3): 2033–2049.

Song, W., and B. Schmeiser. 2009. "Omitting Meaningless Digits in Point Estimates: The Probability Guarantee of Leading-Digit Rules". *Operations Research* 57 (1): 109–117.

Taillard, E. 1993. "Benchmarks for Basic Scheduling Problems". *European Journal of Operational Research* 64 (2): 278–285.

Talwar, P. 1967. "A Note on Sequencing Problems with Uncertain Job Times". *Journal of Operations Research Society of Japan* 9:93–97.

Upasani, A., R. Uzsoy, and K. Sourirajan. 2006. "A Problem Reduction Approach for Scheduling Semiconductor Wafer Fabrication Facilities". *IEEE Transactions on Semiconductor Manufacturing* 19 (2): 216–225.

Vieira, G. E., J. W. Herrmann, and E. Lin. 2003. "Rescheduling Manufacturing Systems: A Framework of Strategies, Policies, and Methods". *Journal of Scheduling* 6 (1): 39–62.

Wang, K., S. Choi, and H. Lu. 2015. "A Hybrid Estimation of Distribution Algorithm for Simulation-Based Scheduling in a Stochastic Permutation Flowshop". *Computers and Industrial Engineering* 90:186–196.

Waschneck, B., T. Altenmüller, T. Bauernhansl, and A. Kyek. 2017. "Production Scheduling in Complex Job Shops from an Industrie 4.0 Perspective: A Review and Challenges in the Semiconductor Industry". CEUR Workshop Proceedings.

AUTHOR BIOGRAPHIES

JOSE M FRAMINAN is a Professor of Industrial Engineering in the School of Engineering at the University of Seville. He has co-authored 80+ refereed publications in the field of decision support systems for operations and supply chain management. He is Editor of the European Journal of Industrial Engineering, and Area Editor for the Flexible Manufacturing and Services Journal. His email address is framinan@us.es.

PAZ PEREZ-GONZALEZ is an Assistant Professor in the School of Engineering at the University of Seville. Her research interests include statistics, simulation and scheduling. He has co-authored a number of papers in different areas of scheduling research. Her email address is pazperez@us.es.

VICTOR FERNANDEZ-VIAGAS is an Assistant Lecturer in in the School of Engineering at the University of Seville. He has co-authored several works related to flowshop scheduling under different objectives and constraints. His email address is vfernandezviagas@us.es.