SCHEDULING IN STOCHASTIC FLOWSHOPS WITH INDEPENDENT SETUP, PROCESSING AND REMOVAL TIMES

Ali Allahverdi
Industrial and Systems Engineering, Department of Mechanical and Industrial Engineering, College of Engineering and Petroleum, Kuwait University, P.O. Box 5969, Safat, 13060, Kuwait

(Received March 1996; in revised form January 1997)

Abstract—The two-machine flowshop problem, where setup and removal times are independent from processing times and machines suffer stochastic breakdowns, is considered with respect to makespan performance measure. After the problem is formulated, an objective function for the makespan is developed. Under certain conditions on the breakdown processes, it is shown that Sule and Huang's [5] algorithm for the deterministic problem stochastically minimizes makespan for the stochastic counterpart problem. © 1997 Elsevier Science Ltd

1. INTRODUCTION

Scheduling $n$ jobs on a two-machine flowshop was a classic problem in the scheduling literature since Johnson [1] introduced the problem. The problem is to find a sequence of the jobs on the machines so that the total time required to complete all $n$ jobs, the so-called makespan, is minimum.

Under certain assumptions, Johnson provided a simple rule by which an optimal sequence can be constructed. These assumptions include (i) setup times depend on processing times, (ii) removal times depend on processing times, and (iii) machines do not suffer breakdowns. Note that when setup and removal times depend on processing times, then these times can be included into the processing times. There exist some environments which do not satisfy these conditions under which case the sequence obtained by Johnson's rule may not be optimal. Sule [2] discussed the environments where assumptions (i) and (ii) are not realistic, while Allahverdi and Mittenthal [3] specified the reasons for considering the environments in which machines are subject to random breakdowns.

Yoshida and Hitomi [4] relaxed assumption (i) for Johnson's problem and provided an optimal algorithm. Sule and Huang [5] relaxed both assumptions (i) and (ii) and specified an optimal algorithm for the problem.

Allahverdi and Mittenthal [3] relaxed assumption (iii) and showed that the Johnson's rule stochastically minimizes makespan. Allahverdi [6] showed that Yoshida and Hitomi's algorithm stochastically minimizes makespan by relaxing assumptions (i) and (iii). The problem which remains unsolved is to relax all the three assumptions (i), (ii), and (iii), which is the topic of this article.

This article considers scheduling on a two-machine flowshop with separated setup, processing, and removal times where machines are subject to random breakdowns. In this article, it is shown that Sule

† (email: allaha@kuc01.kuniv.edu.kw).
‡ Ali Allahverdi is an Assistant Professor in the Department of Mechanical and Industrial Engineering of Kuwait University. He worked for two years an an Assistant Professor in the Industrial Engineering Department of Marmara University, Istanbul, before coming to Kuwait University. He received his B.S. in Petroleum Engineering from Istanbul Technical University and his M.Sc. and Ph.D. in Industrial Engineering, from Rensselaer Polytechnic Institute. His current research interests include scheduling in both stochastic and deterministic environments. He has published in journals such as Computers and Operations Research, Naval Research Logistics, European Journal of Operations Research, The Journal of the Operational Research Society, Mathematical and Computer Modelling, Transactions on Operational Research, and Communications in Statistics: Simulation and Computation.
and Huang's algorithm for the deterministic problem stochastically minimizes makespan for the stochastic counterpart problem provided certain conditions on the breakdowns hold.

2. PROBLEM DESCRIPTION

Consider scheduling \( n \) jobs on a two-machine flowshop such that each job has two operations, where the first operation is done on machine 1 and the second is performed on machine 2. Processing times for all jobs are known and constant, and all the jobs are available for processing at time zero. No splitting of the jobs is allowed. Moreover, job setup and removal times are not included into their processing times, i.e., separated. In other words, the setup and removal times are independent of the processing times. Finally, machines may suffer random breakdowns. The problem is to determine the schedule which stochastically minimizes makespan.

When the setup time is independent of the processing time, the setup on machine 2 can be completed before the job finishes with machine 1 if there exists an idle time on machine 2. If the idle time slot is not enough, at least part of the setup on machine 2 can be completed. Further, when the removal time of the job is also independent of the processing time, then the job is available from machine 1 when its setup and processing on machine 1 is completed; however, machine 1 is not available for the next job until the removal operation on machine 1 is finished. These activities can be seen in Fig. 1 for a typical schedule.

The breakdown process of machine \( k \) can be described by a sequence of non-negative random vectors \( \{ U_r, D_r \} \) for \( r = 1, 2, \ldots \), where \( U_r \) and \( D_r \) denote the \( r \)th uptime and downtime on machine \( k \), respectively. Machine \( k \) is up (working) for some random period of time \( U_r \) and then it breaks down for a random period \( D_r \). After it is fixed, it is up again for another random period \( U_{r+1} \), and then it breaks down for some other random period \( D_{r+1} \), and so on. Associated with the sequence of uptimes, there is a counting process \( \{ N_k(t): t \geq 0 \} \) where \( N_k(t) \) represents the number of breakdowns taking place up to time \( t \) on machine \( k \). The definition of \( N_k(t) \) is given by \( N_k(t) = \sup \{ r \geq 0: \sum_{i=1}^{r} U_i \leq t \} \), where \( U_0 = 0 \). Note that \( t \) remains unchanged for machine \( k \) when either that machine is available but idle, or the machine is down as a result of a breakdown.

We assume that machines are subject to random breakdowns while in operation or being setup or a job is being removed. We also assume that as soon as the second machine has finished processing an operation, that machine is setup for the next operation. We further assume that if a breakdown takes place while an operation of a job is being done, the work done on the job is not lost. We consider simple recourse strategy as opposed to general recourse. Allahverdi [7] briefly describes these two strategies.

Let \( \omega \) denote one realization of uptimes and downtimes, and \( \Omega \) represent the set of all realizations of uptimes and downtimes. Let \( X \) and \( Y \) be two random variables which are defined on the same sample space \( \Omega \). \( X \) is said to be stochastically smaller than \( Y \), denoted as \( X \leq_s Y \), if \( \Pr(X \leq t) \geq \Pr(Y \leq t) \) and stochastically equal to \( Y \), denoted as \( X =_s Y \), if \( \Pr(X \leq t) = \Pr(Y \leq t) \) for all \( t \). The notion of minimizing an objective function stochastically was used by Kijima et al.[8], Pinedo [9], Birge et al. [10], Foley and

![Fig. 1. A typical schedule.](image-url)
Scheduling in stochastic flowshops

Suresh [11,12], Ku and Niu [13], Allahverdi and Mittenthal [3,14] in stochastic scheduling.

A sequence stochastically minimizes an objective function, e.g., makespan, if the makespan of that sequence is stochastically smaller than that of any other sequence.

3. FORMULATION AND THE RESULT

The objective function considered in this article is makespan, denoted by \( R_{\text{max}} \). Once an expression for total idle time on the second machine in terms of job setup, processing and removal times (denoted by \( \Delta \)) is developed, then it is easy to write down an expression for the makespan. The following variables are defined for the development of the objective function:

- \( t_{ik} \): processing time of job \( i \) on machine \( k \),
- \( S_{ik} \): setup time of job \( i \) on machine \( k \),
- \( R_{ik} \): removal time of job \( i \) on machine \( k \),
- \( t_{(i_k)} \): processing time of the job in position \( i \) on machine \( k \),
- \( S_{(i_k)} \): setup time of the job in position \( i \) on machine \( k \),
- \( R_{(i_k)} \): removal time of the job in position \( i \) on machine \( k \),
- \( F_{jk} \): sum of setup, processing, and removal times of the jobs in positions 1,2,...,\( j \) on machine \( k \), i.e.,

\[
F_{jk} = \sum_{r=1}^{j} (S_{rk} + t_{(r_k)} + R_{(r_k)})
\]

\( BD_k(a, b) \): the amount of breakdown in the interval \((a, b)\) on machine \( k \), i.e.,

\[
BD_k(a, b) = \begin{cases} 0 & \text{no breakdowns, i.e., } N_k(b) = N_k(a) \\ \sum_{r=N_k(a)+1}^{N_k(b)} D_r & \text{at least one breakdown, i.e., } N_k(b) > N_k(a). \end{cases}
\]

It can be shown that

\[
R_{\text{max}} = F_{o1} + BD_2(0, F_{o2}) + \Delta
\]

where

\[
\Delta = \max \{0, \Delta_1, \Delta_2, \ldots, \Delta_n\},
\]

and

\[
\Delta_j = F_{j-1,1} + S_{j01} + t_{j01} + BD_1(0, F_{j-1,2} + S_{j02} + t_{j02}) - [F_{j-1,2} + S_{j02} + BD_1(0, F_{j-1,2} + S_{j02})]
\]

for \( j = 1,2,\ldots,n \) with \( F_{01} = F_{o2} = 0 \).

**Lemma**

Assume that \( \{D_{a1}\} \) and \( \{D_{a2}\} \) have identical distributions, and \( N_1(t) \) and \( N_2(t) \) are counting processes having the same distribution. Assume further that both of the counting processes posses the stationary increments property. Given processing lengths \( T_1 \) and \( T_2 \), then \( BD_1(0, T_1) + BD_2(0, T_2) = BD_1(0, T_1 + T_2) \).

**Proof.** Note that since \( \{D_{a1}\} \) and \( \{D_{a2}\} \) have identical distributions, and \( N_1(t) \) and \( N_2(t) \) follow the same distribution, \( BD_1(0, T_2) = BD_2(0, T_2) \). Therefore,

\[
BD_1(0, T_1) + BD_2(0, T_2) = BD_1(0, T_1 + T_2).
\]

But notice that the counting processes posses the stationary increments property. This implies that \( BD_1(0, T_2) = BD_1(T_1, T_1 + T_2) \). Hence,

\[
BD_1(0, T_1) + BD_2(0, T_2) = BD_1(0, T_1) + BD_2(T_1, T_1 + T_2) = BD_1(0, T_1 + T_2).
\]

**Lemma**

Note that a counting process is said to posses stationary increments property if the distribution of the number of breakdowns that occur in any interval of time depends only on the length of the time interval. In other words, the process has stationary increments property if the number of breakdowns in the interval \((t_i + s, t_j + s)\) has the same distribution as the number of breakdowns in the interval \((t_i, t_j)\) for all \( t_i < t_j \) and \( s > 0 \).

Consider the following two sequences of jobs \( \pi_1 \) and \( \pi_2 \). Sequence \( \pi_1 \) has job \( i \) in position \( \tau \) and job
\( h \) in position \( \tau + 1 \). Sequence \( \pi_i \) is obtained from sequence \( \pi_i \) by only interchanging the jobs in positions \( \tau \) and \( \tau + 1 \) so that job \( h \) is in position \( \tau \) and job \( i \) is in position \( \tau + 1 \). Let \( \omega \) denote any realization of uptimes and downtimes and \( \Delta(\pi_k, \omega) \) represent \( \Delta \) for sequence \( \pi_k \) and realization \( \omega \). Then, by definition,

\[
\Delta(\pi_1, \omega) = F_{r-1,1} + S_{i} + t_{i} + BD_1(0, F_{r-1,1} + S_{i} + t_{i}) \\
- [F_{r-1,2} + S_{a} + BD_2(0, F_{r-1,2} + S_{a})],
\]

\[
\Delta(\pi_2, \omega) = F_{r-1,1} + S_{a} + t_{a} + BD_1(0, F_{r-1,1} + S_{a} + t_{a}) \\
- [F_{r-1,2} + S_{a} + BD_2(0, F_{r-1,2} + S_{a})],
\]

\[
\Delta_{+1}(\pi_1, \omega) = F_{r-1,1} + S_{i} + t_{i} + R_{i} + S_{a} + t_{a} + BD_1(0, F_{r-1,1} + S_{i} + t_{i} + R_{i} + S_{a} + t_{a}) \\
- [F_{r-1,2} + S_{a} + BD_2(0, F_{r-1,2} + S_{a})],
\]

\[
\Delta_{+1}(\pi_2, \omega) = F_{r-1,1} + S_{a} + t_{a} + R_{a} + S_{a} + t_{a} + BD_1(0, F_{r-1,1} + S_{a} + t_{a} + R_{a} + S_{a} + t_{a}) \\
- [F_{r-1,2} + S_{a} + BD_2(0, F_{r-1,2} + S_{a})],
\]

Now, consider differences between some of the above pairs. More specifically,

\[
\Delta(\pi_2, \omega) - \Delta(\pi_1, \omega) = t_{a} + S_{a} + t_{a} - (t_{i} + S_{i} + S_{a}) \\
+ BD_1(0, F_{r-1,1} + S_{a} + t_{a}) + BD_2(0, F_{r-1,2} + S_{a}) \\
- [BD_1(0, F_{r-1,1} + S_{i} + t_{i}) + BD_2(0, F_{r-1,2} + S_{a})],
\]

\[
\Delta_{+1}(\pi_2, \omega) - \Delta_{+1}(\pi_1, \omega) = t_{a} + R_{a} + S_{a} - (t_{a} + R_{a} + S_{a}) \\
+ BD_1(0, F_{r-1,1} + S_{i} + t_{i} + R_{i} + S_{a} + t_{a} + BD_2(0, F_{r-1,2} + S_{a} + t_{a} + R_{a} + S_{a}) \\
- [BD_1(0, F_{r-1,1} + S_{i} + t_{i} + R_{i} + S_{a} + t_{a} + BD_2(0, F_{r-1,2} + S_{a} + t_{a} + R_{a} + S_{a})],
\]

Theorem. Consider scheduling on a two-machine flowshop with separate setup, processing, and removal times where both machines suffer random breakdowns. Under the assumptions stated in the Lemma, job \( h \) should precede job \( i \) if:
Scheduling in stochastic flowshops

\[
\min \{ S_h - S_{h2} + t_{h1}, \ t_{h2} + R_{h2} - R_{h1} \} \leq \min \{ S_i - S_{i2} + t_{i1}, \ t_{i2} + R_{i2} - R_{i1} \}
\]
in a sequence which stochastically minimizes makespan.

**Proof.** Note that regardless of the sequence, the term \( F_{h2} \) in (1) is constant, and the term \( BD_{2}(0, F_{h2}) \) in (1) is a random variable whose value is independent of the sequence. Therefore, minimizing \( R_{\text{max}} \) stochastically is equivalent to minimizing \( \Delta \) stochastically. Hence, to show that makespan for a sequence \( \pi \) is stochastically smaller than the makespan for another sequence \( \pi' \), it suffices to show that \( \Delta(\pi, \omega) \leq \Delta(\pi', \omega) \) for any \( \omega \).

For the sequences \( \pi \) and \( \pi' \) specified earlier, it is obvious that \( \Delta(\pi, \omega) = \Delta(\pi', \omega) \) for \( j = 1, 2, \ldots, n \). Hence to show that \( \Delta(\pi, \omega) \leq \Delta(\pi', \omega) \), it suffices to show that;

\[
\max \{ \Delta(\pi, \omega), \Delta_{\pi+}(\pi, \omega) \} \leq \max \{ \Delta(\pi', \omega), \Delta_{\pi+}(\pi', \omega) \}.
\]

Now, observe that the condition;

\[
\min \{ S_{h1} - S_{h2} + t_{h1}, \ t_{h2} + R_{h2} - R_{h1} \} \leq \min \{ S_{i1} - S_{i2} + t_{i1}, \ t_{i2} + R_{i2} - R_{i1} \}
\]
can be considered under two alternatives.

**Alternative 1:** \( \min \{ S_{h1} - S_{h2} + t_{h1}, \ t_{h2} + R_{h2} - R_{h1} \} = S_{h1} - S_{h2} + t_{h1} \), then,

\[
S_{h1} - S_{h2} + t_{h1} \leq S_{i1} - S_{i2} + t_{i1}, \text{ or } S_{h1} + t_{h1} \leq S_{i1} + t_{i1} + S_{i2}.
\]

(6)

and \( S_{h1} - S_{h2} + t_{h1} \leq t_{h2} + R_{h2} - R_{h1} \), or \( S_{h1} - S_{h2} + t_{h1} \leq t_{h2} + R_{h2} + S_{h2} \).

(7)

Note that by (2) and (6),

\[
\Delta(\pi, \omega) \leq \Delta_{\pi+}(\pi, \omega).
\]

(8)

Note also that by (5) and (7),

\[
\Delta_{\pi+}(\pi, \omega) \leq \Delta(\pi, \omega).
\]

(9)

Now by (8) and (9) we have

\[
\max \{ \Delta(\pi, \omega), \Delta_{\pi+}(\pi, \omega) \} \leq \max \{ \Delta(\pi, \omega), \Delta_{\pi+}(\pi, \omega) \}.
\]

Alternative 2: \( \min \{ S_{h1} - S_{h2} + t_{h1}, \ t_{h2} + R_{h2} - R_{h1} \} = t_{h2} + R_{h2} - R_{h1} \), then,

\[
t_{h2} + R_{h2} - R_{h1} \leq S_{h1} - S_{h2} + t_{h1}, \text{ or } t_{h2} + R_{h2} + S_{h2} \leq S_{h1} + R_{h1} + t_{h1},
\]

(10)

and \( t_{h2} + R_{h2} - R_{h1} \leq t_{h2} + R_{h2} - R_{h1} \), or \( t_{h2} + R_{h2} + S_{h2} \leq t_{h2} + R_{h2} + R_{h1} \).

(11)

Observe that by (4) and (10),

\[
\Delta(\pi, \omega) \leq \Delta_{\pi+}(\pi, \omega).
\]

(12)

Observe also that by (3) and (11),

\[
\Delta_{\pi+}(\pi, \omega) \leq \Delta_{\pi+}(\pi, \omega).
\]

(13)

Now by (12) and (13) we have

\[
\max \{ \Delta(\pi, \omega), \Delta_{\pi+}(\pi, \omega) \} \leq \max \{ \Delta(\pi, \omega), \Delta_{\pi+}(\pi, \omega) \}.
\]

For both alternatives we have shown that \( \max \{ \Delta(\pi, \omega), \Delta_{\pi+}(\pi, \omega) \} \leq \max \{ \Delta(\pi, \omega), \Delta_{\pi+}(\pi, \omega) \} \). Therefore, \( \Delta(\pi, \omega) \leq \Delta(\pi, \omega) \). Since this argument is valid for any \( \omega \), \( \Delta(\pi, \omega) \leq \Delta(\pi, \omega) \). Hence, in a sequence which stochastically minimizes makespan, job \( h \) should precede job \( i \). This completes the proof.

Now, a sequence which stochastically minimizes makespan can be constructed by Sule and Huang’s algorithm which is as follows.

1. Step 1: Find the minimum value among the values of \( (S_{h1} - S_{h2} + t_{h1}) \) and \( (t_{h2} + R_{h2} - R_{h1}) \) \( (h=1, 2, \ldots, n) \).
2. Step 2: If the minimum is \( (S_{h1} - S_{h2} + t_{h1}) \), place job \( h \) first, if the minimum is \( (t_{h2} + R_{h2} - R_{h1}) \), place job \( h \) last.
3. Step 3: Remove the assigned job from consideration and if there are unscheduled jobs go to Step 1, otherwise stop.
4. CONCLUDING REMARKS

This article has addressed the two-machine flowshop problem, where setup times and removal times are independent from processing times and machines suffer random breakdowns, with the performance measure of makespan. In this article, it was shown that Sule and Huang’s algorithm stochastically minimizes makespan under some assumptions on the breakdown processes.

There might be some flowshops where these assumptions are not valid. It may be impossible to find an optimal sequence without making any assumption on the breakdown processes. However, without any assumptions on the breakdown processes, some dominance rules might be established, by which only a partial sequence can be found. It may also be of interest to investigate the case where one machine stochastically dominates the other, i.e., when the expected breakdown duration in a given period of processing on one machine is larger than that of the other. Note that under the conditions assumed in this article, the expected breakdown duration for a given period is the same for both machines.

This problem of scheduling in flowshops with independent setup, processing, and removal times has been only addressed with respect to the performance measure of makespan (Sule [2], and Sule and Huang [5] considered the deterministic environment, the present paper has considered the stochastic environment). Some other performance measures such as mean flowtime and maximum lateness are also of interest. Thus, this problem might also be addressed with regard to these performance measures for both the deterministic and stochastic environments.

REFERENCES