Int J Flex Manuf Syst (2006) 17:201–226
DOI 10.1007/s10696-006-8819-1

A heuristic approach to minimize expected makespan in open shops subject to stochastic processing times and failures

David Alcaide · Andrés Rodriguez-Gonzalez · Joaquín Sicilia

© Springer Science + Business Media, LLC 2006

Abstract Many real world situations exist where job scheduling is required. This is the case of some entities, machines, or workers who have to execute certain jobs as soon as possible. Frequently what happens is that several workers or machines are not available to perform their activities during some time periods, due to different circumstances. This paper deals with these situations, and considers stochastic scheduling models to study these problems. When scheduling models are used in practice, they have to take into account that some machines may not be working. That temporal lack of machine availability is known as breakdowns, which happen randomly at any time. The times required to repair those machines are also random variables. The jobs have operations with stochastic processing times, their own release times, and there is no precedence between them. Each job is divided into operations and each operation is performed on the corresponding specialized machine. In addition, in the problems considered, the order in which the operations of each job are done is irrelevant. We develop a heuristic approach to solve these stochastic open-shop scheduling problems where random machine breakdowns can happen. The proposed approach is general and it does not depend on the distribution types of the considered random input data. It provides solutions to minimize the expected makespan. Computational experiences are also reported. The results show that the proposed approach gives a solid performance, finding suitable solutions with short CPU times.

D. Alcaide (✉) · A. Rodriguez-Gonzalez · J. Sicilia
Departamento de Estadística, Investigación Operativa y Computación
Universidad de La Laguna, Tenerife, Canary Islands, Spain
e-mail: dalcaide@ull.es

A. Rodriguez-Gonzalez
e-mail: arguezg@ull.es

J. Sicilia
e-mail: jsicilia@ull.es

Keywords Stochastic open-shop scheduling · Expected makespan · Random breakdowns · Random repair times

1. Introduction

Open-shop scheduling problems are characterized by a set of parts of different types, which have to be executed by a set of specialized machines. Each part is divided into different operations. Each of these operations is performed on its corresponding specialized machine. The order in which the operations of each job are done is irrelevant. In stochastic open-shop scheduling problems, some input data are not fixed but are random variables. Also, times where breakdowns happen are not known in advance. Usually these problems appear in industrial processes, good manufacturing systems, management and scheduling of people assistance services, information services with several specialized servers, timetable scheduling problems, applicants or candidate selection processes, customer attendance in bank or postal offices, and people attendance in public offices (tourism information offices, town information offices, government information offices, town management offices, and government management offices).

There are many of these practical situations where it might be sufficient to outline and solve these problems using deterministic models. In the literature we find several references concerning these deterministic open-shop scheduling problems. Among them, are the books of Baker (1974), French (1982), Blazewicz et al. (2001), and Brucker (2001), and the papers of Lawler et al. (1993) and Anderson et al. (1997). An important well-known algorithm to solve these deterministic models is the algorithm of Gonzalez and Sahni (1976). This algorithm solves optimally with $O(n)$ complexity the 2-machine deterministic open-shop scheduling problem, where the objective is to minimize makespan, namely $O2 \,||\, C_{\max}$. However, the general case with three or more machines is NP-hard (Gonzalez et al., 1976); Lawler et al. (1981, 1982, 1993).

Nevertheless, many other problems require stochastic models for their study. For example, in management and scheduling of people attention and/or information services, the following problems are contemplated. The people attention unit could be, for example, a bank branch, a department store, a town information office, a candidate selection committee. This people attention unit has m specialized attendants/servers. Those specialized attendants/servers could be, for example, bank clerks, shop assistants and/or department store managers, connoisseurs or experts in different kinds of specific town information, qualified and professional specialist committee members. These specialized attendants/servers are devoted to attending/serving n users. These users are, for example, business-persons and/or clients, customers, tourists, citizens, companies, employers, job applicants and/or grant applicants or other kinds of applicants or candidates. Each user (job) has its own release time and demands several specialized and specific information/services (the operations of the job). These operations have stochastic processing times. Thus, these problems could be studied using stochastic open-shop scheduling models.

The stochastic open-shop problems are usually difficult. They belong to the NP-hard computational complexity problems class. As far as we know, in the scheduling literature, several authors have studied stochastic scheduling problems. Pinedo's book (2002) focuses on both the theory and the applications of scheduling including stochastic scheduling problems. Different general aspects about stochastic scheduling problems, together with models and algorithms to solve them, can also be found in, among others, the papers of Pinedo and Schrage (1982) and Weiss (1982, 1995). Moreover, Pinedo and Rammouz (1988) and Li and Cao (1995) analyze single machine stochastic scheduling problems subject to random breakdowns. Scheduling problems on non-specialized parallel machines, where breakdowns are allowed, are contemplated by Cai and Zhou (1999) and Allahverdi and Mittenthal (1994b). Pinedo (1981) deals with stochastic job-shop problems. Stochastic flow-shop and job-shop problems without random breakdowns are analyzed by Pinedo and Wie (1986). Several papers by Allahverdi and collaborators are devoted to flow-shop problems with random breakdowns. Some of them consider two-machine flow-shop problems (see Allahverdi, 1995, 1996, 1997, 1999; Allahverdi and Mittenthal, 1994a, 1995, 1998). Other papers study proportionate flow-shop problems (Allahverdi, 1996; Allahverdi and Savsar 2001). Flow-shop problems, where random breakdowns are considered and processing times are assumed known and constant (non-random variables), are studied by Allahverdi (1995, 1996, 1997, 1999) and Allahverdi and Mittenthal (1994a, 1995, 1998). General stochastic flow-shop scheduling problems subject to breakdowns are studied by Alcaide et al. (2002), where a general procedure to convert problems with breakdowns into problems without breakdowns is proposed. Pinedo and Ross (1982) consider stochastic open-shop scheduling problems without breakdowns. They provide an optimal algorithm to minimize the expected makespan in the two-machine case, when all jobs are available at time zero, and the processing times of the operations follow exponential random distributions, which do not depend on the machines.

In contrast to previous papers, the present paper deals with stochastic open-shop scheduling problems subject to random breakdowns. The aim in these problems is to minimize the expected makespan. In addition, the problems analyzed are not constrained to a limited number of machines or certain processing time distributions, i.e., the proposed approach depends on neither the number of machines nor the processing time distributions.

The paper is organized as follows: Section 2 introduces notation and formulates the problem. Section 3 studies the computational complexity, calculates the distribution function of the remaining processing times, and analyzes the computation of the expected makespan. These results are general and applicable for any processing time random distributions. Section 4 provides a heuristic algorithm to solve the considered problem. A numerical example is presented in Section 5. Computational experience is shown in Section 6. Finally, conclusions are given and further research is commented on.

2. Problem formulation

It is well known that the makespan open-shop deterministic scheduling problem can be defined in the following terms. There are n jobs to be executed by m machines. The machines are specialized in such a way that any job J_j, $j = 1, \ldots, n$ is divided into m operations O_{ij}, $i = 1, \ldots, m$, and each operation is performed on its corresponding specialized machine. The order in which the operations of a job are done on the machines is irrelevant. Then, without loss of generality, it is possible to number machines and operations in such a way that operation O_{ij} is the operation of job J_j to be performed on machine M_i, with processing time p_{ij}, $i = 1, \ldots, m$, $j = 1, \ldots, n$. Job J_j is available at its release time r_j, $j = 1, \ldots, n$. Jobs are independent and the aim is to minimize the makespan.

In the case of some problem data being random variables, we have stochastic open-shop scheduling problems. This paper studies these situations considering, in addition, the fact that random breakdowns happen and job preemption is technically possible to deal with these random breakdowns. In these preemption situations, preemption-resume mode is considered, i.e., the performance of the preempted job can be continued at the point where the job was preempted, without repeating any part of it (see Baker, 1974, page 81, for details about preemption modes). More precisely, let X_{ij} be the processing time of operation O_{ij} of job J_j performed on machine M_i. That time is a random variable with distribution function $F_{X_{ij}}$. Such operation O_{ij} can be preempted after T processed time units. In this case, and taking into account preemption resume mode, the remaining processing time of the operation O_{ij} is the random variable $X^*_{ij}(T) = (X_{ij} - T)|\{X_{ij} \geq T\}$, with distribution function $F_{X^*_{ij}(T)}$. Note that $X^*_{ij}(0) = X_{ij}$. We will denote the random variable $X^*_{ij}(T)$ by X^*_{ij} when there is no confusion. Other random variables involved in the stochastic problems considered in this paper are described in the following paragraphs.

The time period length when machine M_i is available, that is, the time between two consecutive breakdowns on M_i is measured with the random variable Y_i, F_{Y_i} being its distribution function, $i = 1, \ldots, m$. For machine M_i, the random variable Y_i could change in the course of time due to causes such as the age or aging of machine M_i or, even, due to improvements and updates done on machine M_i. For these situations we can consider in the model several random variables $Y_{i\tau}$, $i = 1, \ldots, m$, $\tau = 1, 2, \ldots$, where the random variable $Y_{i\tau}$, with distribution function $F_{Y_{i\tau}}$, measures the time between $(\tau - 1)$-th and τ-th breakdowns of machine M_i.

In the same way, the length of the repair time intervals could be evaluated using random variables. The required time to repair machine M_i is measured with the random variable Z_i, F_{Z_i} being its distribution function, $i = 1, \ldots, m$. Also in this case, for machine M_i, it is possible that the random variable Z_i changes in the course of time due to the age or aging of machine M_i, the improvements and updates done on machine M_i or, even, due to developments, updates and improvements in the used technical repair procedures. These situations are considered in the model with the random variables $Z_{i\tau}$, $i = 1, \ldots, m$, $\tau = 1, 2, \ldots$, where the random variable $Z_{i\tau}$, with distribution function $F_{Z_{i\tau}}$, measures the necessary time to repair the τ-th breakdown on machine M_i. In this way, we model the fact that, as time goes on, the distribution of the necessary repair times also changes.

Note that, if $Y_{i\tau} = Y_i$, $\tau = 1, 2, \ldots$, and/or if $Z_{i\tau} = Z_i$, $\tau = 1, 2, \ldots$, we have several particular cases where the length of the available or without-breakdowns periods of machines and/or the required repair times, respectively, always follow the same probability distribution.

Let Σ be the set of feasible solutions. That is, for any schedule $\sigma \in \Sigma$, σ satisfies the release times r_j, $j = 1, \ldots, n$, and the basic conditions of the scheduling problems are also verified. These basic conditions are: the impossibility of performing the same job simultaneously in two or more machines, and the prohibition of two or more jobs being performed on the same machine at the same time.

Given $\sigma \in \Sigma$, $C_j(\sigma)$ is the random variable which measures the completion time of job J_j in schedule σ. Thus, we are interested in finding a schedule $\sigma^* \in \Sigma$ with a minimum expected completion time for all the jobs, that is, with the minimum expected makespan. Therefore, we must schedule jobs on machines in such a way that we can expect to finish all jobs as soon as possible. Thus, our problem is

$$\min\{E[C_{\max}(\sigma)]/\sigma \in \Sigma\} \qquad (1)$$

where, for each schedule $\sigma \in \Sigma$, the random variable which assess the makespan is

$$C_{\max}(\sigma) = \max_{j=1,\ldots,n} \{C_j(\sigma)\} \qquad (2)$$

Therefore, we deal with stochastic open-shop scheduling problems subject to random breakdowns and with random repair times, where the objective is to minimize the expected completion time of all the jobs. The paper considers problems where all jobs are available at time zero ($r_j = 0, \forall j$, static problems) and problems with different release times (dynamic problems). Both problems have maximum computational complexity as is analyzed in the following section.

3. Problem analysis and theoretical results

To study stochastic open-shop scheduling problems subject to random breakdowns we use a general approach or performing guide, given in Alcaide et al. (2002), that allows us to decide dynamically during the course of time. This general procedure converts any scheduling problem with m machines ($m \geq 1$) subject to breakdowns into a set of scheduling problems with several machines which are not subject to breakdowns. The sets of machines in the corresponding free-breakdowns scheduling problems are subsets of the set of machines associated with the original problem subject to breakdowns. The mentioned conversion is applicable to both specialized machines problems and non specialized machines problems. The conversion can be used whether all jobs are available at time zero or if the release times of the jobs are different. It is also valid when the problem is stochastic or deterministic, and for any problem with one or several machines.

The aforementioned general procedure studies the problem subject to breakdowns by solving a sequence of problems without breakdowns. In fact, the general procedure decides in each time, taking into account all the available information up to such instant, and without information about the next decision time. The approach keeps its decision until a new decision instance arises.

So, this procedure is a dynamic procedure that makes decisions in the course of time. In each time t, let $m(t)$ be the number of working or non-broken down machines, with $0 \leq m(t) \leq m$. The set $S(t)$ of non-broken machines at time t and its cardinal $m(t)$ is the information that determines the system status at time t. During the course of time, the status of the system can be modified at time $t_k \geq t$ if at least one of the following situations occurs: new breakdowns happen, or new release times r_j are achieved, or the repair of some broken down machine finishes. In this way, in the course of time,

$t_0, t_1, \ldots, t_k, \ldots$ will arise where breakdowns occur, new release times are achieved, or new repaired machines come back. These times will identify the different intervals or scheduling periods $I_1 =]t_0, t_1]$, $I_2 =]t_1, t_2]$, $\ldots$, $I_k =]t_{k-1}, t_k] \ldots$. In each one of these intervals, $m(t)$ has a constant value that is usually, but not necessarily, different from the $m(t)$-value in the previous and successive intervals (i.e. $m(t) = m_k$ constant $\forall t \in I_k$, and usually, but not necessarily, $m_k \neq m_{k+1}$, $k = 1, 2, \ldots$). Moreover, the set S_k of working or non-broken down machines is constant in each of these intervals I_k, but it changes from one interval to the consecutive interval ($S_k \neq S_{k+1}$, $k = 1, 2, \ldots$). Due to the way they are defined, within the intervals I_k, it is possible that neither new breakdowns occur nor the repairs of broken down machines are finished. That is, each period I_k, $k = 1, 2, \ldots$ is a free-breakdowns period and no repair finishes inside period I_k.

In this way the original problem with breakdowns is converted into a sequence of problems without breakdowns. These without-breakdown problems are solved in the intervals $I_1, I_2, \ldots$ making decisions at the decision instances $t_0, t_1, t_2, \ldots$. These decision times are the lower extremes of the above mentioned intervals. The decisions are chosen at the starting of each time interval without any information about the end of the interval. The decision is kept during the interval until such interval ends. In any interval I_k, the associated without-breakdowns problem is characterized by the set S_k of non-broken machines (remember this set is unchangeable in the period I_k), and by the set $\mathcal{U}_k$ of uncompleted jobs at the starting time of interval I_k, with $n_k = card(\mathcal{U}_k)$. Each job $j \in \mathcal{U}_k$ is characterized by those operations O_{ij} which have not been finished in previous intervals. These scheduling problems without breakdowns must be solved using the available methods in the literature, when these methods exist.

It is important to perceive that, at the beginning of each free-breakdown scheduling period $I_k =]t_{k-1}, t_k]$, $k = 1, 2, \ldots$, we know the value t_{k-1} that marks the starting time of the period I_k, but we do not know when t_k will appear. This t_k will mark the end of the period I_k when either the current working machines set S_k changes, or when new release times r_j are achieved. However, at time t_{k-1}, we know the set of working machines S_k for the period I_k. Thus, at t_{k-1}, we must solve the associated without-breakdowns scheduling problem using only the available information at t_{k-1}. That is, we have to schedule the uncompleted jobs without knowing t_k. Consequently, we keep the decided schedule for the period I_k while there are more jobs to schedule and while the system status does not change.

The study of the computational complexity of the stochastic open-shop scheduling problems is reduced to the study of the computational complexity of the corresponding without-breakdowns problems. This analysis is done in the following subsection. The other two subsections are devoted to providing a theoretical formula to compute the distribution functions of the remaining processing times, and to commenting on several aspects for computing the expected makespan.

3.1. Problem complexity

This subsection analyzes the computational complexity of the stochastic open-shop scheduling problems. Both problems where all jobs are available at time zero ($r_j = 0$, $j = 1, \ldots, n$) and problems where jobs have different release times are analyzed.

When all jobs are available at time zero, the original problem (1) with random breakdowns is reduced to stochastic open-shop scheduling problems but without breakdowns. At each time t_{k-1}, it is necessary to solve only one of these without-breakdowns problems, characterized by the set S_k of m_k working machines, $m_k = card(S_k)$, and the set $\mathcal{U}_k$ of uncompleted jobs (or jobs parts) at such instant t_{k-1}.

If the number m_k of available machines is 1, then any schedule without idle time minimizes the expected makespan and solves the corresponding without-breakdowns problem in scheduling period $I_k =]t_{k-1}, t_k]$ (with t_k unknown). If $m_k = 2$, Pinedo and Ross (1982) found an optimal schedule with minimum expected makespan for this stochastic without-breakdowns static open-shop scheduling problem when processing times are drawn from exponential distributions. The optimal schedule requires that these distributions do not depend on machines, i.e., the problem solved by Pinedo and Ross is $O2|X_{ij} \sim exp(\lambda_j)|E[C_{\max}]$ according to an extended notation of the three parameters classification initially proposed by Graham et al. (1979). However, when $m_k = 2$, it appears to be very difficult to generalize the Pinedo and Ross' method for any class of distributions for processing times (see Pinedo, 2002). Finally, if $m_k \geq 3$, as far we know, there are no optimal efficient algorithms to solve this stochastic open-shop scheduling problem. Furthermore, the following theorem can be established.

Theorem 1. *The stochastic open-shop scheduling problem (1) without breakdowns, with release times $r_j = 0$, $j = 1, \ldots, n$, considering general distributions of processing times X_{ij}, and any number m of machines, is NP-hard.*

Proof: Using an extended notation of the three parameters classification initially proposed by Graham et al. (1979), we are going to prove that the problem denoted by $O|X_{ij} \sim G_{ij}|E[C_{\max}]$, in the cases of general distributions of processing times X_{ij} and any number m of machines, is NP-hard. It is known that the deterministic problem $O||C_{\max}$, in the general case, is already NP-hard (Gonzalez and Sahni, 1976, 1978). Also, due to the deterministic problem being a particular case of the problem $O|X_{ij} \sim G_{ij}|E[C_{\max}]$ when the distributions G_{ij} are degenerated in the constants distributions $G_{ij} \sim p_{ij}$, and taking into account that the deterministic problem is NP-hard, it follows that $O|X_{ij} \sim G_{ij}|E[C_{\max}]$ is NP-hard. □

Corollary 1. *The stochastic open-shop scheduling problem (1) with breakdowns, with release times $r_j = 0$, $j = 1, \ldots, n$, considering general distributions of processing times X_{ij}, and any number m of machines, is NP-hard.*

Proof: Trivial by Theorem 1 and because any scheduling problem with breakdowns can be converted into problems without breakdowns. □

Next, let us consider the case where release times r_j, $j = 1, \ldots, n$ are distinct. It is clear that scheduling problems with different release times are always more difficult than the corresponding scheduling problems with all jobs available at time zero. Then, it follows that

Corollary 2. *The stochastic open-shop scheduling problem (1) without breakdowns, where release times are distinct, considering general distributions of processing times*

X_{ij}, and any number m of machines, is NP-hard. In addition, if random breakdowns happen, the problem is also NP-hard.

Proof: The proof in the free-breakdowns case is obvious taking into account Theorem 1, and due to scheduling problems with different release times being more difficult than those corresponding to all jobs available at time zero (see also Graham et al., 1979; Lawler et al., 1982).

When random breakdowns happen, the proof is also trivial due to Corollary 1 and the same reasonings for release times. □

The next subsection provides a theoretical formula to compute the distribution functions of the remaining processing times of the operations of the uncompleted jobs.

3.2. Distribution functions of the remaining processing times

The processing time of the operation O_{ij} of job J_j performed on machine M_i is measured by the random variable X_{ij} with distribution function $F_{X_{ij}}$. Such operation could be preempted after T processing time units are performed. Following preemption resume mode, the remaining processing time of such operation is the random variable $X^*_{ij}(T)$. If there is no confusion, we can denote this random variable by X^*_{ij}. The theoretical formula to compute the distribution functions of these random variables is:

$$\begin{aligned} F_{X^*_{ij}(T)}(x^*) &= P(X^*_{ij} \le x^*) \\ &= P(X_{ij} - T \le x^* | X_{ij} \ge T) \\ &= \frac{P(X_{ij} - T \le x^*, X_{ij} \ge T)}{P(X_{ij} \ge T)} \\ &= \frac{P(T \le X_{ij} \le x^* + T)}{1 - P(X_{ij} < T)} \\ &= \frac{F_{X_{ij}}(x^* + T) - F^-_{X_{ij}}(T)}{1 - F^-_{X_{ij}}(T)}. \end{aligned} \quad (3)$$

With this formula it is possible to compute the distribution function of the remaining processing times X^*_{ij} for any distribution of the processing times X_{ij}. So, for example, if the processing time X_{ij} follows the exponential distribution $X_{ij} \sim exp(\lambda)$, then $F_{X_{ij}}(x) = 1 - e^{-\lambda x}$. Thus, using the above formula (3), we have $F_{X^*_{ij}}(y) = 1 - e^{-\lambda y}$ and the remaining processing time X^*_{ij} also follows the exponential distribution $X^*_{ij} \sim exp(\lambda)$. If the processing time X_{ij} follows the uniform distribution $X_{ij} \sim U[a, b]$, then $F_{X_{ij}}(x) = \frac{x-a}{b-a}$. Thus, by (3), $F_{X^*_{ij}(T)}(y) = \frac{y}{b-T}$ and the remaining processing time X^*_{ij} also follows the uniform distribution $X^*_{ij} \sim U[0, b - T]$. The computation in the case of any other distribution will be done in a similar way using the formula (3).

In the next subsection theoretical aspects for computing the expected makespan are commented on.

3.3. Expected makespan

Taking into account that random breakdowns can happen, the non-broken machines set changes as time goes on. These changes determine the different free-breakdown scheduling periods $I_k =]t_{k-1}, t_k]$, $k = 1, 2, \ldots$. In addition, at the beginning of each scheduling period $I_k =]t_{k-1}, t_k]$ we know t_{k-1} which marks the starting time of the period I_k, but we do not know when t_k will appear. This t_k will mark the end of the period I_k with a change in the current working machines set S_k. This fact implies that it is impossible to calculate directly from the initial instant $t_0 = 0$ the expected makespan $E[C_{\max}(\sigma)]$, with $\sigma \in \Sigma$.

Note that we know neither how many breakdowns will happen in a feasible solution $\sigma \in \Sigma$ nor when these breakdowns will appear. Thus, given $\sigma \in \Sigma$, the mean of the random variable $C_{\max}(\sigma)$, defined in (2), i.e. the expected makespan, $E[C_{\max}(\sigma)]$, will not be directly calculable from the initial instant $t_0 = 0$.

However, at t_{k-1} we know the set of non-broken machines S_k for the period I_k. Thus, at t_{k-1}, we must solve the associated without-breakdowns scheduling problem using only the available information at that time. We can know the distribution of the random variables: remaining processing times (X_{ij}^*) (using Eq. (3)), working time intervals $(Y_{i\tau})$, and breakdowns repair times $(Z_{i\tau})$. Nevertheless, we have to schedule without knowing t_k. So, we have to keep the decided schedule for the period I_k while there are jobs to schedule, while new release times r_j are not achieved, and while the non-broken machines set does not change. The system status will change at t_k, which marks the final of the period I_k. So, the beginning and ending of the different scheduling periods I_k are detected during the course of time. Also, the free-breakdown schedules in these periods are found, and the expected makespan will be the expected value of the ending time of the last scheduling period.

4. A heuristic approach

In this section, we present a heuristic approach to solve open-shop stochastic scheduling problems subject to random breakdowns. Note that, at the beginning, we have a set S_1 of non-broken machines. This set changes as time goes on due to breakdowns happening, newly repaired machines coming back, or new release times r_j being achieved because new jobs appear. The machines are specialized and it is necessary to perform n jobs $J_1, \ldots, J_n$ on them. We distinguish the problems where all jobs are available at time zero ($r_j = 0$, $j = 1, \ldots, n$) from problems where jobs have different release times.

4.1. The case where all jobs are available at time zero

We have to decide at any instance t_{k-1}, $k = 1, 2, \ldots$ a schedule for the corresponding open-shop stochastic scheduling problem without breakdowns. Then, we have to keep this schedule during the period $I_k =]t_{k-1}, t_k]$, that is, while there are jobs to complete and until the working machines set changes. Note that t_k is unknown. Time t_k will be known only when the current non-broken machines set S_k changes, due to new breakdowns happening or new available machines coming.

Nevertheless, at time t_{k-1}, we know the set S_k of current non-broken machines, the value $m_k = card(S_k)$, the set $\mathcal{U}_k$ of jobs and/or jobs parts which are not completed at time t_{k-1}, and the value $n_k = card(\mathcal{U}_k)$. We have to solve an open-shop stochastic scheduling problem without breakdowns with m_k machines and n_k jobs. This partial problem could be denoted as $Om_k||E[C_{\max}]$. In this way, the original open-shop stochastic scheduling problem subject to random breakdowns is converted into a sequence of open-shop stochastic scheduling problems without breakdowns. At the end of the general procedure, when $\mathcal{U}_k = \emptyset$, in the last scheduling period I_k, the final solution σ is built merging the solutions found in the different scheduling periods.

It is important to remark that the known algorithms for solving, in an exact way, these stochastic open-shop without-breakdowns problems (which appear in the time intervals I_k, $k = 1, 2, \ldots$), are usually limited to a determined number of machines, or these algorithms consider certain processing time distributions of the jobs (see Pinedo, 2002; Pinedo and Ross, 1982). So, if the number of working machines in the current free-breakdowns period I_k is $m_k = 1$, any feasible solution without idle-time is optimal. If $m_k = 2$ and the remaining processing times follow exponential distributions which do not depend on the machines, we can use Pinedo and Ross' algorithm (Pinedo and Ross, 1982) to solve optimally the corresponding without-breakdowns stochastic open-shop scheduling problem in period I_k.

However, many stochastic open-shop scheduling problems without breakdowns are still open, i.e., there is no efficient optimal algorithms to solve them. When these problems appear, while we are solving the original stochastic open-shop problem with breakdowns, we need to use alternative ways to solve it. One way is to obtain approximate solutions by solving the deterministic open-shop scheduling problem considering the processing times $p_{ij} = E[X^*_{ij}]$. These deterministic problems are of the form $O||C_{\max}$. When there are $m = 2$ machines, this deterministic problem could be solved in $O(n)$ time by Gonzalez and Sahni algorithm (Gonzalez et al., 1976). Nevertheless, when the number of machines is $m \geq 3$, this deterministic problem is NP-hard (Gonzalez et al., 1976; Lawler et al., 1981; 1982; 1993) and heuristic approaches are necessary to solve it. In this case the chosen heuristic approach is the tabu search algorithm given in Alcaide, Sicilia, and Vigo, (1997).

The next subsection concerns the case where jobs have different release times.

4.2. The case where jobs have different release times

When there are different release times, we have to take into account that the problems to solve in the successive scheduling periods I_k are different and more complicated. In period I_k, an open-shop stochastic scheduling problem without breakdowns, with m_k machines, and where jobs have different release times, must be solved. This partial problem could be denoted as $Om_k|r_j|E[C_{\max}]$. As far as we know, there is no way of solving optimally these stochastic problems in polynomial time. Even if the remaining processing times X^*_{ij} follow exponential distributions, the problem is very difficult.

Taking into account these difficulties to solve open-shop stochastic problems with random breakdowns and different release times, we propose an approach based on the transformation of the original problem with random breakdowns and different release times into a sequence of stochastic open-shop problems without breakdowns. The corresponding free-breakdowns scheduling periods $I_k =]t_{k-1}, t_k]$ are shortened

```
Approximate Algorithm
/* Initialization */
   Put $t_0 = 0$, $k = 1$;
   Let $S_k$ be the set of non-broken machines at time $t_{k-1}$;
   Let $\mathcal{U}_k$ be the set of available jobs (or jobs parts) still uncompleted at time $t_{k-1}$;
   /* (at $t_0 = 0$, $\mathcal{U}_1 = \{$ jobs $J_j$ with $r_j = 0\}$) */
   Let $m_k = card(S_k)$;
   while $\mathcal{U}_k \neq \emptyset$ {
/* Schedule at time $t_{k-1}$ (with $t_k$ unknown) */
      if ($m_k = 1$) {
         Choose any schedule without idle-time, because that schedule minimizes the expected makespan;
      }
      if ($m_k = 2$ and $X^*_{ij} \sim exp(\lambda_j)$) {
         Use Pinedo and Ross algorithm (1982) to find an optimal solution;
      }
      if ($m_k = 2$ and $X^*_{ij}$ follow other distributions) {
         Consider $p_{ij} = E[X^*_{ij}]$;
         Apply Gonzalez and Sahni algorithm (1976) to solve the corresponding deterministic problem;
         /* (it provides an approximate solution to the stochastic problem) */
      }
      if ($m_k \geq 3$) {
         Consider $p_{ij} = E[X^*_{ij}]$;
         Apply tabu search algorithm of Alcaide et al. (1997)
         to solve the corresponding deterministic problem;
         /* (it provides an approximate solution to the stochastic problem) */
      }
      Keep the proposed solution at time $t_{k-1}$ until:
         (a) any breakdown happens in one (or more) of the current working machine(s), or
         (b) newly repaired machine(s) comes back, or
         (c) new times $r_j$ are achieved, i.e., new jobs appear to be performed;
      Let $t_k$ be the first time when (a) or (b) or (c) happens;
      /* Scheduling period $I_k = ]t_{k-1}, t_k]$ finishes and starts a new one */
      Put $k = k + 1$;
      Update $S_k$ and $m_k = card(S_k)$;
      Update the set $\mathcal{U}_k$ of available jobs (or jobs parts) still uncompleted at $t_{k-1}$;
      Put $n_k = card(\mathcal{U}_k)$;
   }
STOP
```

Fig. 1 Approximate algorithm structure

to avoid different release times inside I_k. In each scheduling period I_k, a without-breakdown problem with all jobs available at time zero is solved in the same way already described in Section 4.1.

The structure of this approximate algorithm is shown in Fig. 1. The problem to solve at time t_{k-1} is a $Om_k||E[C_{\max}]$ scheduling problem, i.e., considering the set S_k of non-broken machines and the set $\mathcal{U}_k$ of available jobs or job parts. Note that, at the end of the algorithm, when $\mathcal{U}_k = \emptyset$, in the last scheduling period I_k, the final solution σ is formed by combining the solutions found in the different scheduling periods.

To end this section, it is important to remark that both procedures (either the case when all jobs are available at time zero or the case when jobs have different release times) must dynamically decide during the course of time. That is done using all the available information at each decision instance t_{k-1}, with t_k unknown. The decision is kept until t_k arises (cases (a) or (b) or (c) in algorithm showed in Fig. 1). In any case, in each instance t_{k-1} (with t_k unknown) the algorithm chooses the most suitable possible decision with the available information at such time.

To illustrate the algorithm, a numerical example is presented in the next section. In this example we solve the problem for both cases: considering $r_j = 0, \forall j$, and when different release times exist for the jobs.

5. An open-shop scheduling example

Let us consider an open-shop scheduling example where $n = 5$ jobs must be performed on $m = 3$ machines. First, we consider the static case, i.e., all jobs are available at time zero ($r_j = 0, j = 1, \dots, n$), and later, we will analyze the same example considering different release times.

Note that the proposed approach decides in accordance with the evolution of the system. The approach keeps its decision until changes arise in the system. These changes are described in the approximate algorithm structure (see Fig. 1). We do not know the evolution of the system, and we need to decide taking into account this evolution. Hence, we have to simulate the random variables which characterize that evolution.

With these remarks, the input data of this stochastic problem are the following ($i = 1, 2, 3$ and $j = 1, 2, 3, 4, 5$):

1. Processing times X_{ij} have exponential distributions of parameters λ_j independent of the machines, i.e., $E[X_{ij}] = 1/\lambda_j$ are taken randomly from uniform distribution in [10, 20]. Thus, after simulation, we have the mean vector

$$(E[X_{ij}]) = (1/\lambda_j) = (10.44, 11.34, 15.05, 10.54, 19.36).$$

2. Working time interval lengths Y_i have exponential distributions, where the means $E[Y_i]$ are taken randomly from uniform distribution in [20, 50]. Thus, after simulation, we have the mean vector

$$(E[Y_i]) = (25.16, 32.40, 45.54).$$

3. Repair times Z_i have exponential distributions, where the means $E[Z_i]$ are taken randomly from uniform distribution in [10, 20]. Thus, after simulation, we have the mean vector

$$(E[Z_i]) = (12.63, 18.15, 16.44).$$

Using the approximate algorithm given in Section 4, we provide an approximate solution for this stochastic open-shop scheduling problem subject to random breakdowns. The algorithm has to decide dynamically at the start of each scheduling period. Then, the algorithm keeps its decision until (a) all the jobs have been completed or (b) the current scheduling period finishes. To see when (a) or (b) happen, we have to observe the evolution of the real system. If we do not know the evolution of the system, we have to simulate it.

So, the evolution of the system in this example has been simulated taking into account the previous random distributions of processing times, the availability times and the repair times. Processing times obtained by simulation are:

$$(X_{ij}) = \begin{pmatrix} 15.11 & 11.97 & 18.30 & 2.27 & 29.51 \\ 13.73 & 37.26 & 7.18 & 16.59 & 4.18 \\ 4.25 & 5.31 & 47.63 & 11.04 & 13.84 \end{pmatrix}$$

and the sequences of working and repair times are:

1. Machine 1: $Y_{11} = 23.55$, $Z_{11} = 6.96$, $Y_{12} = 38.86$, $Z_{12} = 13.93$, $Y_{13} = 26.87$, $Z_{13} = 7.12, \dots$
2. Machine 2: $Y_{21} = 5.99$, $Z_{21} = 10.26$, $Y_{22} = 39.99$, $Z_{22} = 8.02$, $Y_{23} = 40.14$, $Z_{23} = 25.91, \dots$
3. Machine 3: $Y_{31} = 72.50$, $Z_{31} = 5.10$, $Y_{32} = 25.63$, $Z_{32} = 23.25$, $Y_{33} = 50.26$, $Z_{33} = 18.29, \dots$

Note that different events of the evolution of the system are not known until they effectively happen. So, the first time that machine 1 breaks down is at 23.55 time units, next, it is 6.96 time units being repaired, and so on.

Thus, the development of the algorithm is as follows:

First scheduling period: We have $k = 1, t_0 = 0, I_1 =]0, t_1]$, with t_1 unknown. Schedule at $t_0 = 0$ with the available information at that instance: $S_1 = \{M_1, M_2, M_3\}$, $\mathcal{U}_1 = \{1, \dots, 5\}$. We have a $O3|X_{ij} \sim exp(\lambda_j)|E[C_{\max}]$ scheduling problem. It is heuristically solved by considering $p_{ij} = E[X_{ij}]$ and solving the corresponding deterministic problem with the tabu search algorithm provided by Alcaide et al. (1997). This solution is kept during this scheduling period. In this case, on M_1 the jobs must follow the sequence $\{4, 3, 2, 1, 5\}$, on M_2 the sequence of jobs is $\{5, 4, 3, 2, 1\}$ and on M_3 the sequence is $\{1, 2, 5, 4, 3\}$. Taking into account the evolution (simulated) of the system the scheduling period I_1 finishes at time $t_1 = \min\{Y_{11} = 23.55, Y_{21} = 5.99, Y_{31} = 72.50\} = 5.99$. At that time, machine M_2 breaks down, scheduling period I_1 finishes, and a new scheduling period starts.

We have scheduled, in this period, on machine M_1 jobs 4 and 3, on machine M_2 jobs 5 and 4, and on machine M_3 jobs 1 and 2 up to the time 5.99 (see Fig. 2). At $t_1 = 5.99$ the machine M_2 breaks down while it performs job 4. This job is unfinished on M_2 and its remaining part is sent to the next scheduling period. Note that we can calculate the distribution of the remaining processing times X^*_{13}, X^*_{24}, and X^*_{32} of the operations of the jobs 3, 4, and 2 on machines 1, 2, and 3, respectively, using the formula (3). As processing times follow exponential distributions, then, the remaining part of job 4 on machine M_2 is sent to the next scheduling period as a new job with exponential processing time. These considerations are the same for job 3 on machine M_1, and for job 2 on machine M_3. Note that, at $t_1 = 5.99$, the corresponding operations of job 4 on machine M_1, of job 5 on machine M_2, and of job 1 on machine M_3 are already completed.

Second scheduling period: Now, we have $k = 2$, $t_1 = 5.99$, $I_2 =]5.99, t_2]$, with t_2 unknown. We have to schedule the remaining jobs at $t_1 = 5.99$ using the available information at that instance: $S_2 = \{M_1, M_3\}, \mathcal{U}_2 = \{1, 2, 3, 4, 5\}$ (note that the corresponding operations of job 4 on machine M_1, of job 5 on machine M_2, and of job

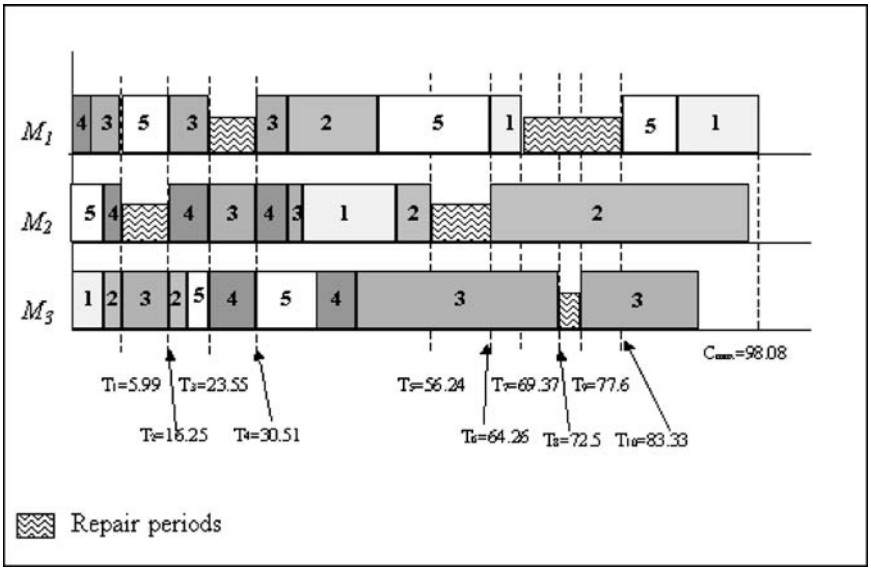

Fig. 2 Open-shop numerical example when all jobs are available at time zero

1 on machine M_3 are already completed). We have a $O2|X^*_{ij} \sim exp(\lambda_j)|E[C_{\max}]$ scheduling problem and we can apply the Pinedo and Ross' algorithm (Pinedo and Ross, 1982) to find an optimal solution in the current scheduling period $I_2 =]5.99, t_2]$. Pinedo and Ross' algorithm uses the list of jobs in a non-increasing order of expected processing times, in this case $\{5, 3, 2, 4, 1\}$, to assign successively the jobs to the first available and free machine, i.e., job 5 is assigned to machine M_1 and job 3 to M_3 and so on. This solution is kept until the ending time t_2 of such scheduling period arises. This happens at $t_2 = \min\{Y_{11} = 23.55, Y_{21} + Z_{21} = 5.99 + 10.26, Y_{31} = 72.50\} = 16.25$ (see Fig. 2.)

Third scheduling period: In this period we have $k = 3$, $t_2 = 16.25$, $I_3 =]16.25, t_3]$, with t_3 unknown. We have to schedule at $t_2 = 16.25$ using the available information at that instance: $S_3 = \{M_1, M_2, M_3\}$, $\mathcal{U}_3 = \{1, 2, 3, 4, 5\}$. Notice that the corresponding operations of job 4 on machine M_1, of job 5 on machine M_2, and of job 1 on machine M_3 are already completed. We have a $O3|X^*_{ij} \sim exp(\lambda_j)|E[C_{\max}]$ scheduling problem. By considering $p_{ij} = E[X^*_{ij}]$, and solving the corresponding deterministic problem by applying the tabu search algorithm of Alcaide et al. (1997), we have the following schedule: on M_1 the jobs follow the sequence $\{3, 2, 5, 1\}$, on M_2 the sequence of jobs is $\{4, 3, 1, 2\}$, and on M_3 the sequence is $\{2, 5, 4, 3\}$. Following the evolution of the system, this scheduling period I_3 finishes at time $t_3 = 23.55$ when machine M_1 breaks down.

Continuing in this way, at the end of eleventh period, I_{11}, we have a final solution with makespan 98.08 (see Fig. 2.)

Let us consider the same numerical example above, but with the only difference being that jobs have different release times. Then, the input data are the same and, in addition, we have the following vector of release times $(r_j) = (10, 0, 0, 20, 30)$.

Thus, the development of the algorithm is as follows:

First scheduling period: We have $k = 1, t_0 = 0, I_1 =]0, t_1]$, with t_1 unknown. Schedule at $t_0 = 0$ with the available information at that instance: $S_1 = \{M_1, M_2, M_3\}, \mathcal{U}_1 = \{2, 3\}$. We have a $O3|X_{ij} \sim exp(\lambda_j)|E[C_{\max}]$ scheduling problem. It is heuristically solved by considering $p_{ij} = E[X_{ij}]$ and solving the corresponding deterministic problem with the tabu search algorithm provided by Alcaide et al. (1997). This solution is kept during this scheduling period. In this case, on M_1 the jobs must follow the sequence $\{3, 2\}$, on M_2 the sequence of jobs is $\{3, 2\}$ and on M_3 the sequence is $\{2, 3\}$. Taking into account the evolution (simulated) of the system the scheduling period I_1 finishes at time $t_1 = \min\{Y_{11} = 23.55, Y_{21} = 5.99, Y_{31} = 72.50\} = 5.99$. At this instance, machine M_2 breaks down, and scheduling period I_1 finishes.

In that period $I_1 =]0, 5.99]$ we should schedule job 3 on machine M_1, job 3 on machine M_2, job 2 on machine M_3. However, due to job 3 being performed on M_1 and job 2 being performed on machine M_3, no jobs are performed on machine M_2 until some of them finish. As job 2 finishes on M_3 at time 5.31, then, at this instant job 2 can start its processing on machine M_2 and it continues up to time 5.99 (see Fig. 3). At $t_1 = 5.99$ the machine M_2 breaks down. Job 3 is unfinished on M_1 and job 2 is unfinished on M_2. Then, the remaining parts of these jobs are sent to the next scheduling period. Using the formula (3), we can calculate the remaining processing times X_{13}^* and X_{22}^*. As processing times follow exponential distributions, then, the remaining parts of jobs 3 and 2 are sent to the next scheduling period as new jobs with exponential processing times. Note that, at $t_1 = 5.99$, the corresponding operation of job 2 on machine M_3 is already completed.

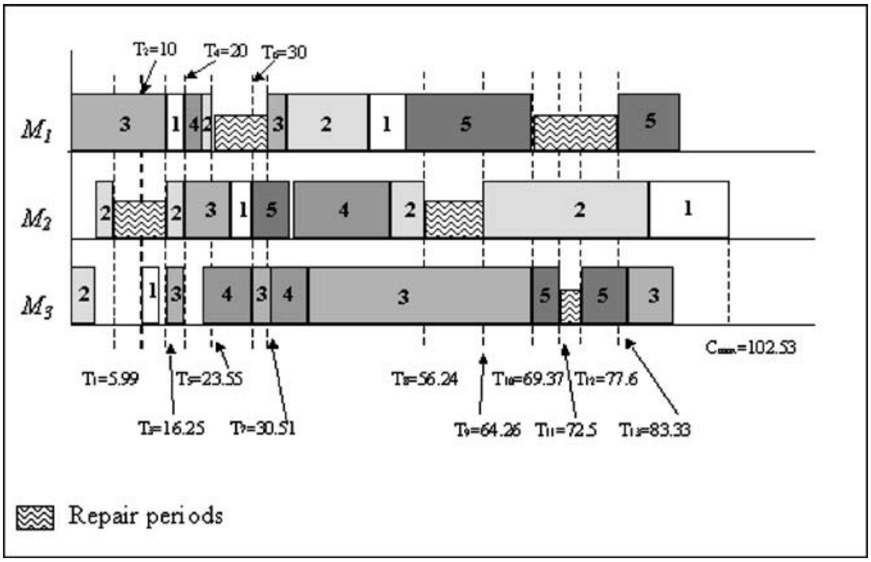

Fig. 3 Open-shop numerical example when jobs have different release times

Second scheduling period: Now, we have $k = 2$, $t_1 = 5.99$, $I_2 =]5.99, t_2]$, with t_2 unknown. We have to schedule the remaining jobs at $t_1 = 5.99$ using the available information at that instantce: $S_2 = \{M_1, M_3\}$, $\mathcal{U}_2 = \{2, 3\}$. Note that the corresponding operation of job 2 on machine M_3 is already completed. We have a $O2|X^*_{ij} \sim exp(\lambda_j)|E[C_{\max}]$ scheduling problem and we can apply the Pinedo and Ross' algorithm (Pinedo and Ross, 1982) to find an optimal solution from the starting time $t_1 = 5.99$ in the current scheduling period $I_2 =]5.99, t_2]$. Pinedo and Ross' algorithm uses the list of jobs in a non-increasing order of expected processing times (in this case the ordered list is $\{3, 2\}$). The jobs are successively assigned to the first available working machine, i.e., job 3 is assigned to machine M_1 and no job is assigned to M_3 (because job 3 is being processed on M_1 and job 2 is already finished on M_3). This solution is kept until the ending time t_2 of such scheduling period arises. This happens at $t_2 = \min\{r_1 = 10, Y_{11} = 23.55, Y_{21} + Z_{21} = 5.99 + 10.26, Y_{31} = 72.50\} = 10$ (see Fig. 3).

Third scheduling period: In this period, $k = 3$, $t_2 = 10$, $I_3 =]10, t_3]$, with t_3 unknown. We have to schedule at $t_2 = 10$ using the available information at that instantce: $S_3 = \{M_1, M_3\}$, $\mathcal{U}_3 = \{1, 2, 3\}$. Notice that the corresponding operation of job 2 on machine M_3 is already completed. We have a $O2|X^*_{ij} \sim exp(\lambda_j)|E[C_{\max}]$ scheduling problem and we apply the Pinedo and Ross' algorithm (Pinedo and Ross, 1982) to find an optimal solution from the starting time $t_2 = 10$ in the current scheduling period $I_3 =]10, t_3]$. The Pinedo and Ross algorithm uses the list of jobs $\{3, 2, 1\}$ to assign the jobs successively to the first working and available machine, i.e., job 3 is assigned to machine M_1, and job 1 is assigned to M_3 (remember job 2 is already finished on M_3). This solution is kept until the ending time t_3 of such scheduling period arises. This happens at $t_3 = \min\{r_4 = 20, Y_{11} = 23.55, Y_{21} + Z_{21} = 5.99 + 10.26, Y_{31} = 72.50\} = 16.25$ (see Fig. 3).

Fourth scheduling period: We have $k = 4$, $t_3 = 16.25$, $I_4 =]16.25, t_4]$, with t_4 unknown. It is necessary to schedule at $t_3 = 16.25$ using the available information at that instance: $S_4 = \{M_1, M_2, M_3\}$, $\mathcal{U}_4 = \{1, 2, 3\}$. Notice that the corresponding operations of jobs 1 and 2 on machine M_3 are already completed. We have a $O3|X^*_{ij} \sim exp(\lambda_j)|E[C_{\max}]$ scheduling problem. It is heuristically solved by considering $p_{ij} = E[X^*_{ij}]$ and solving the corresponding deterministic problem with the tabu search algorithm of Alcaide et al. (1997). This solution is kept during this scheduling period. In this case, on M_1 the jobs must follow the sequence $\{1, 2, 3\}$, on M_2 the sequence of jobs is $\{1, 2, 3\}$, and on M_3 the sequence is $\{2, 3, 1\}$. So, job 1 is assigned to machine M_1, job 2 to machine M_2 (because job 1 is being processed on M_1), and job 3 is assigned to M_3 (because job 2 is already finished on M_3). This solution is kept until the ending time t_4 of such scheduling period arises. This happens at $t_4 = \min\{r_4 = 20, Y_{11} = 23.55, t_3 + Y_{22} = 16.25 + 39.99, Y_{31} = 72.50\} = 20$ (see Fig. 3).

Continuing in this way, at the end of fourteenth period, I_{14}, we have a final solution with makespan 102.53 (see Fig. 3).

In the next section a computational experience is described.

6. Computational experience

We have employed a AMD Athlon XP 1.7 computer (1.40 GHz and 512 Mb RAM) to analyze the computational performance of the heuristic approach. The C programming language has been used to program the algorithm.

This experience is done through simulations of the evolution of the real system. In these simulations we have considered that the number m of machines is 3, 5 and 10, the number n of jobs is 5, 10, 25, 50 and 100, and, for simplicity, the release times are $r_j = 0\, \forall j$.

For each simulation the test instances set is divided into 27 classes according to the possibility of considering the variables X (processing times), Y (length of working intervals), Z (repair times) as short (S), medium (M) or long (L). The classification of the random variables X, Y, Z into the classes S, M, L depends on the mean of the random variable. Table 1 shows the correspondences.

As an example, SML is a XYZ-selection in which processing times are short, times where the machines are working, are medium and repair times are long. SLL and SLS are two XYZ-selections that differ only because SLL has long repair times and SLS has short repair times. In the same way, SLM and SSM are two XYZ-selections that differ only because SLM has long working times and SSM has short working times.

The heuristic algorithm given in Section 4 does not depend on the distribution types of the random input data. The proposed approach decides dynamically in accordance with the evolution of the system, and it keeps its decision until changes arise in the system. These changes are due to new breakdown(s) happening, or new repaired machines coming back, or new jobs appearing. Also note that we do not know the evolution of the system, and we need to decide taking into account this evolution. This is the reason for simulating the random variables which characterize the unknown evolution in our computational experience.

With these remarks, and for simplicity, in our computational experience we have considered processing times X_{ij} from the exponential distribution, $X_{ij} \sim exp(\lambda_j)$, where the means $1/\lambda_j$ are taken from the uniform distribution in the intervals specified in Table 1. In this way, we can apply the optimal algorithm of Pinedo and Ross (1982) in the free-breakdowns scheduling periods I_k, $k = 1, 2, \ldots$ when the number of non-broken machines in such periods is $m_k = 2$. Also, length Y_i of working time intervals and the repair times Z_i have exponential distributions, where the respective means $E[Y_i]$ and $E[Z_i]$ are taken randomly from uniform distributions in the intervals provided by Table 1.

Note that the generated data could be likened to a real-life situation, for example, the management and scheduling problems of people attention and/or information services commented on in Section 1. Also, it is known that exponential distributions play an important role in the analysis of the reliability of equipment, in manufacturing

Table 1 Intervals for the means of the random variables X, Y, Z

	S	M	L
X	[1, 10]	[10, 20]	[20, 50]
Y	[10, 20]	[20, 50]	[50, 100]
Z	[1, 10]	[10, 20]	[20, 50]

processes and other industrial models (see, for example, Fabrycky et al. 1972; Grant and Leavenworth, 1996).

In each simulation, we generate s instances for any combination m, n and XYZ-selection. The number of instances s considered is 10, 20, 50, 100 and 200. These instances are analyzed using the general procedure proposed in Section 3. In each period I_k, the partial without-breakdowns problems are solved using different algorithms depending on the number m_k of available machines. This is done in accordance with the approximate algorithm structure summarized in Fig. 1. So, if $m_k = 1$, any schedule without idle-time is optimal. If $m_k = 2$, and taking into account the distributions of processing times X^*_{ij}, we use the Pinedo and Ross' algorithm (1982) to find an optimal solution. Finally, if $m_k \geq 3$, we use the algorithm provided by Alcaide et al. (1997) to find an approximate solution. The results collected in Tables 2–4 correspond to the sample size $s = 50$, i.e., we have 50 instances for each combination m, n and XYZ-selection. In addition, Fig. 4 shows the standard deviation of the makespan for different sample sizes in several combinations of the parameters m, n, and XYZ-selection. Varying the sample size s in the set $10, 20, 50, 100, 200, 500$, the results show us that, for all the m, n, and XYZ-selections, the standard deviation decreases when the sample size increases. Several of these cases are illustrated in Fig. 4.

The values in these tables are average results for the 50 instances solved for each combination m, n and XYZ-selection. The columns are respectively, the number of jobs, the XYZ-selection, the expected makespan, the standard deviation of the random variable $C_{\max}$, the stages or scheduling periods, and CPU times in seconds.

We observe that CPU-times are insignificant in a lot of cases. In fact, for instances with $m = 3$ machines, in only one combination of the input parameters (selection $(m, n, X, Y, Z) = (3, 100, L, S, S)$) the average CPU time of the 50 solved instances is greater than 8 CPU seconds, and there is no combination in which its average time is greater than 11 CPU seconds. For $m = 5$, only seven selections need more than 60 CPU seconds and, among them, only one selection uses more than three minutes CPU time. For $m = 10$, the algorithm is also fast, in fact, for 103 of the 135 selections, i.e., the 76, 29%, take less than one minute, and nearly 83, 70% of the selections take less than two minutes. Only one of the 135 selections takes more than four CPU minutes.

We can also compare the results among different XYZ-selections to see the expected makespan variation. As an example, when the results for XSZ are compared with the corresponding XLZ results, we observe that the expected makespan decreases. This reduction is more significative for some m and n combinations than others. Thus, we can affirm that, in all cases, but in some more than others, it could be advantageous for the decision-maker to invest in renewing his/her machines in such a way that the working times change from short (S) to long (L). The manager will make the decision taking into account the investment cost and the advantages that he/she will get due to the expected reduction of the makespan.

The experience allows us to obtain similar conclusions comparing the results among XYL and XYS selections. So, it provides valid information to decide if it is profitable or not to invest in improving the technical methods and tools destined to repair breakdowns.

Also, note that this kind of computational experience could be useful for studying the sensibility of the expected makespan not only with respect to the X, Y and Z

Table 2 Average results for different open-shop stochastic scheduling problems subject to breakdowns, with $n = 5$ and 10 jobs

No. of jobs	XYZ-selection	Number of machines											
		3				5				10			
		$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time
5	S S S	44.99	2.42	19.92	0.00	27.67	2.68	20.08	0.00	56.01	7.82	92.36	0.06
	S S M	16.55	2.74	9.52	0.00	20.19	1.50	10.20	0.00	64.72	3.04	80.26	0.02
	S S L	42.23	6.86	13.00	0.00	67.56	46.29	27.56	0.00	79.50	4.29	66.42	0.02
	S M S	14.78	0.79	7.94	0.00	19.36	0.25	7.08	0.00	50.01	1.19	41.84	0.04
	S M M	46.33	4.87	17.68	0.00	26.18	1.45	9.14	0.00	51.74	1.12	33.80	0.04
	S M L	25.23	1.86	7.88	0.00	40.79	0.26	10.04	0.00	60.90	3.23	33.18	0.02
	S L S	10.13	0.30	2.00	0.00	17.83	0.14	4.08	0.00	50.14	1.01	15.84	0.02
	S L M	10.13	0.30	2.00	0.00	16.56	1.47	4.08	0.00	49.28	0.02	15.86	0.02
	S L L	10.13	0.30	2.00	0.00	19.62	3.38	4.08	0.00	52.81	5.20	15.10	0.00
	M S S	87.86	0.26	49.08	0.02	130.21	3.34	134.74	0.08	182.85	2.74	413.90	0.38
	M S M	145.74	23.32	49.42	0.00	221.10	29.93	118.86	0.04	196.74	7.04	253.18	0.14
	M S L	179.58	39.25	43.84	0.00	166.17	7.77	67.36	0.04	301.16	9.15	252.98	0.12
	M M S	114.06	4.84	32.20	0.00	103.33	0.82	52.20	0.02	188.99	2.54	203.42	0.18
	M M M	136.07	4.37	33.80	0.02	138.68	2.10	45.10	0.00	180.39	3.99	140.46	0.08
	M M L	121.35	29.62	22.16	0.02	201.58	2.28	52.04	0.00	211.84	7.77	127.64	0.08
	M L S	87.20	20.54	17.58	0.02	101.74	0.67	20.08	0.00	190.11	2.32	80.12	0.10
	M L M	100.95	5.29	15.84	0.00	105.95	0.77	16.14	0.00	187.05	3.49	68.22	0.06
	M L L	121.07	35.71	16.26	0.00	123.96	0.68	17.08	0.00	179.97	5.08	56.20	0.06
	L S S	201.71	1.81	115.96	0.04	262.96	2.51	287.56	0.18	396.01	3.57	914.26	0.84
	L S M	224.41	7.72	73.66	0.02	367.02	5.85	194.82	0.06	463.42	15.52	614.68	0.34
	L S L	494.23	192.70	84.50	0.00	851.63	41.41	279.20	0.06	647.36	23.03	548.84	0.24
	L M S	204.06	7.06	51.84	0.02	228.12	1.01	124.26	0.08	404.96	3.93	454.78	0.50
	L M M	297.67	11.56	69.44	0.00	274.55	2.04	96.18	0.04	374.21	17.79	305.14	0.22
	L M L	212.12	33.45	32.50	0.00	353.41	1.11	87.06	0.02	448.19	25.26	279.22	0.16
	L L S	264.23	2.56	40.08	0.00	224.54	1.54	50.16	0.02	437.82	1.84	206.12	0.28
	L L M	226.60	4.79	25.84	0.00	224.81	2.11	42.24	0.02	393.64	7.40	158.42	0.14
	L L L	265.67	8.92	27.82	0.00	298.88	4.00	47.96	0.02	398.11	15.90	136.82	0.10

(*Continued on next page.*)

Table 2 (*continued*)

		Number of machines											
		3				5				10			
No. of jobs	*XYZ*-selection	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time
10	S S S	18.10	0.90	7.14	0.00	28.46	0.89	22.34	0.00	54.07	2.62	95.70	0.16
	S S M	28.49	6.75	8.46	0.00	40.86	0.42	20.10	0.02	61.02	0.17	74.66	0.06
	S S L	35.60	11.26	5.84	0.00	46.98	3.72	16.24	0.02	99.89	0.14	80.74	0.06
	S M S	18.56	0.92	7.96	0.00	27.99	0.85	10.16	0.00	50.10	1.40	51.36	0.10
	S M M	23.85	1.08	5.96	0.00	36.36	0.49	10.14	0.00	47.03	0.33	32.68	0.04
	S M L	73.56	36.48	9.32	0.00	43.23	3.92	10.24	0.00	57.61	1.72	29.92	0.04
	S L S	16.21	0.89	5.92	0.00	27.22	0.39	4.08	0.02	55.49	2.26	19.72	0.04
	S L M	23.93	0.21	5.92	0.00	28.70	1.83	4.12	0.00	46.56	0.33	14.84	0.04
	S L L	38.84	2.34	5.92	0.00	32.12	4.52	4.18	0.00	54.80	2.55	14.90	0.02
	M S S	146.83	4.12	72.28	0.06	265.00	8.93	280.88	0.38	319.06	4.70	726.96	2.70
	M S M	272.49	7.15	82.24	0.02	268.94	0.27	151.02	0.18	417.25	7.18	538.30	1.04
	M S L	588.22	283.74	112.00	0.00	347.95	8.91	131.28	0.12	475.78	8.13	370.60	0.52
	M M S	118.99	1.91	36.04	0.02	260.27	9.33	123.64	0.18	298.94	4.89	341.70	1.44
	M M M	163.96	4.45	26.70	0.00	300.05	5.82	110.92	0.12	360.80	4.54	298.96	0.82
	M M L	207.17	36.75	26.50	0.02	311.53	4.02	77.06	0.08	449.19	4.94	265.28	0.52
	M L S	139.23	0.87	17.96	0.02	260.91	9.52	47.96	0.08	287.41	1.94	141.22	0.64
	M L M	151.33	4.60	14.04	0.00	287.24	10.03	46.98	0.06	311.60	3.60	129.30	0.48
	M L L	177.11	0.98	12.04	0.00	285.64	5.08	41.02	0.06	338.35	1.55	117.48	0.36
	L S S	308.46	4.39	150.24	0.08	391.27	56.14	459.10	0.68	748.37	3.17	1455.86	5.78
	L S M	456.03	27.39	140.20	0.02	631.01	1.13	362.10	0.30	900.11	9.54	1166.78	2.16
	L S L	973.08	107.21	180.92	0.00	896.22	49.58	335.98	0.20	1207.96	4.69	939.12	1.04
	L M S	297.70	5.45	84.04	0.08	409.80	0.58	204.28	0.30	713.90	18.16	833.38	3.00
	L M M	357.86	13.20	55.56	0.04	602.24	6.24	233.70	0.22	731.36	0.83	609.58	1.70
	L M L	494.78	99.39	72.62	0.02	615.82	0.70	157.22	0.14	930.48	5.02	551.36	1.06
	L L S	287.09	3.81	34.00	0.04	543.14	17.56	112.74	0.12	706.32	15.29	360.20	1.36
	L L M	317.54	18.25	28.54	0.02	573.39	16.55	104.80	0.12	680.66	8.72	293.88	1.00
	L L L	383.83	0.67	26.00	0.02	601.64	10.28	96.94	0.08	707.26	0.21	254.50	0.74

Table 3 Average results for open-shop stochastic scheduling problems subject to breakdowns, with $n = 25$ and 50 jobs

		Number of machines											
		3				5				10			
No. of jobs	*XYZ*-selection	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time
25	S S S	47.75	2.08	18.28	0.02	115.11	1.71	117.24	0.66	196.03	7.09	417.96	2.34
	S S M	104.79	3.82	31.80	0.02	134.72	1.37	72.10	0.28	153.78	2.28	171.02	0.90
	S S L	119.83	27.62	29.04	0.02	133.06	48.65	42.46	0.18	290.90	37.47	213.50	0.70
	S M S	49.44	0.91	10.20	0.02	99.83	12.00	53.20	0.30	146.80	3.31	140.68	0.68
	S M M	43.97	5.90	4.66	0.00	100.87	2.71	28.74	0.14	120.64	0.48	69.16	0.52
	S M L	109.90	8.79	12.24	0.00	179.14	79.79	42.44	0.18	199.68	3.64	91.82	0.62
	S L S	72.34	1.15	4.16	0.02	86.11	3.80	14.00	0.12	123.59	3.75	39.76	0.24
	S L M	40.45	4.37	2.10	0.02	89.79	2.00	12.90	0.08	133.89	2.50	33.92	0.26
	S L L	40.58	5.28	2.10	0.02	86.41	9.00	12.04	0.12	158.70	4.07	33.92	0.34
	M S S	329.03	9.08	155.78	0.18	490.95	40.18	470.20	5.60	868.02	17.67	1672.32	13.18
	M S M	642.38	64.12	185.46	0.10	729.05	0.36	398.46	1.00	1029.60	6.30	1283.54	18.72
	M S L	775.43	24.51	156.60	0.04	1857.42	225.51	607.48	0.68	1469.90	67.68	1155.72	7.56
	M M S	400.44	11.20	125.68	0.12	399.88	16.47	219.48	1.90	663.74	3.54	679.62	15.60
	M M M	361.15	12.29	50.44	0.04	594.03	7.79	215.64	1.00	795.08	21.03	636.70	9.48
	M M L	616.28	163.62	83.54	0.02	1033.79	21.35	246.44	0.84	989.49	31.80	577.18	6.66
	M L S	312.12	4.84	28.28	0.02	410.72	37.01	91.24	1.24	685.43	15.71	316.60	7.32
	M L M	323.11	7.28	24.16	0.02	506.59	10.51	96.30	0.96	707.64	6.28	265.38	5.10
	M L L	398.64	16.70	24.38	0.02	596.74	6.85	95.00	0.84	790.92	27.09	264.52	4.32
	L S S	839.01	1.49	404.08	0.40	1223.22	18.67	1290.14	9.08	2014.87	63.73	3917.48	30.12
	L S M	931.87	52.92	260.88	0.16	1544.07	36.63	834.22	2.44	1971.87	26.63	2437.88	12.30
	L S L	2770.68	254.64	555.20	0.08	3747.05	141.55	1146.12	1.54	3175.91	51.47	2480.06	7.78
	L M S	967.67	32.87	245.68	0.22	938.95	5.52	454.12	3.66	1335.90	19.21	1402.90	15.18
	L M M	744.68	25.92	103.90	0.10	1331.65	53.04	504.62	2.14	1599.71	28.84	1225.26	8.70
	L M L	1886.44	103.11	275.80	0.08	2114.08	50.31	507.52	1.68	2015.36	27.51	1155.00	5.96
	L L S	758.83	1.35	76.32	0.08	1008.59	40.59	255.36	2.96	1437.94	5.38	654.38	6.70
	L L M	698.46	14.12	48.40	0.08	1012.50	35.41	197.76	1.90	1460.43	20.45	575.16	5.26
	L L L	816.26	8.01	67.86	0.06	1211.90	28.42	200.50	1.08	1588.14	18.23	528.88	3.78

(Continued on next page)

Table 3 *(Continued)*

		Number of machines											
		3				5				10			
No. of jobs	*XYZ*-selection	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time
50	S S S	122.20	5.84	63.98	0.38	220.01	2.41	208.56	4.02	241.92	0.66	477.30	18.66
	S S M	203.65	39.67	59.34	0.14	221.88	5.79	116.68	1.20	359.86	3.99	444.92	10.40
	S S L	335.30	98.30	66.54	0.08	410.96	9.14	144.88	0.92	679.07	34.45	486.64	10.16
	S M S	120.37	5.69	39.78	0.20	163.24	7.16	69.44	0.92	258.21	2.04	249.96	6.30
	S M M	160.94	1.99	23.96	0.16	205.23	2.15	66.34	0.58	250.39	8.22	190.66	6.44
	S M L	217.16	79.04	28.50	0.12	270.47	7.10	67.30	0.46	353.29	1.49	204.30	4.26
	S L S	151.16	5.96	21.92	0.10	162.04	1.00	24.08	0.68	228.50	6.08	88.76	2.34
	S L M	148.04	3.00	15.80	0.10	176.80	3.79	28.24	0.42	222.88	7.62	70.88	2.38
	S L L	144.59	3.50	13.84	0.10	223.12	39.65	25.06	0.76	248.19	2.29	69.42	2.38
	M S S	772.57	24.04	370.24	0.66	1001.60	15.48	973.40	27.16	1271.46	16.68	2679.36	135.08
	M S M	1026.31	55.12	273.30	0.30	1386.57	46.40	772.24	4.68	2095.39	4.08	2626.96	80.20
	M S L	2388.09	270.47	434.48	0.18	2240.05	60.59	814.66	3.70	3085.87	7.97	2587.65	79.61
	M M S	700.84	41.86	187.90	0.74	995.71	7.37	500.90	10.44	1380.66	20.94	1584.92	106.78
	M M M	855.51	30.09	171.42	0.46	1128.82	21.99	397.90	6.56	1447.99	24.68	1084.44	48.56
	M M L	1098.69	70.33	163.02	0.36	1646.00	10.96	433.20	4.12	2017.25	0.64	1089.54	42.52
	M L S	706.45	21.69	92.36	0.46	873.22	14.67	183.70	4.22	1156.60	27.20	538.94	37.66
	M L M	810.21	91.61	75.50	0.16	1033.47	14.32	188.76	3.44	1329.02	5.63	546.78	36.14
	M L L	805.37	22.07	58.20	0.18	1149.51	59.69	178.02	3.18	1530.77	18.31	543.38	29.28
	L S S	1835.66	24.13	821.00	1.64	2285.42	5.05	2243.16	39.56	3211.64	254.20	4865.80	193.34
	L S M	2114.14	93.79	550.16	1.56	3163.44	57.84	1784.28	19.98	4562.45	260.21	4903.11	187.86
	L S L	4794.93	855.86	870.98	0.52	4504.87	563.55	1632.34	10.46	6564.42	359.15	5060.72	92.86
	L M S	1849.70	44.45	450.08	0.70	2592.03	95.09	1469.08	24.30	2778.50	274.34	2834.44	177.90
	L M M	1960.94	18.77	292.72	0.88	2470.58	52.76	863.90	14.22	3146.47	100.17	2516.86	119.94
	L M L	2526.68	346.15	310.12	0.70	3365.76	52.06	869.10	9.96	4170.65	225.62	2453.22	87.46
	L L S	1493.73	66.12	205.20	0.48	1950.16	32.54	429.20	9.98	2574.31	43.86	1235.36	82.60
	L L M	1698.85	63.56	155.10	0.52	2439.18	17.19	480.60	7.56	2734.42	62.47	1126.92	65.84
	L L L	1779.66	84.87	120.50	0.50	2445.01	55.25	386.66	7.38	3032.40	96.54	1060.18	51.00

Table 4 Average results for open-shop stochastic scheduling problems subject to breakdowns, with $n = 100$ jobs

		Number of machines											
		3				5				10			
Number of jobs	*XYZ*-selection	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time	$E(C_{\max})$	$C_{\max}$ deviation	Stages (ave.)	CPU time
100	S S S	343.00	4.60	161.38	1.80	411.90	3.29	408.78	17.50	446.31	7.28	896.46	91.16
	S S M	516.01	2.76	150.60	0.30	492.31	27.30	267.72	7.76	730.40	8.64	888.96	63.50
	S S L	706.29	57.31	118.86	0.18	713.47	14.08	250.68	4.88	1161.27	95.06	857.74	42.86
	S M S	232.58	13.76	48.84	0.60	328.53	27.84	173.32	8.18	413.11	3.80	410.58	44.54
	S M M	376.66	47.27	71.68	0.60	386.66	28.97	136.78	4.84	544.10	2.66	410.30	33.16
	S M L	485.36	40.60	64.32	0.46	487.68	1.36	125.84	3.98	714.45	28.98	384.34	27.72
	S L S	253.20	6.81	22.40	0.26	295.44	4.37	64.20	3.38	393.82	7.41	167.46	16.74
	S L M	343.06	18.11	32.50	0.24	343.05	1.49	63.56	2.62	424.66	7.03	157.42	14.26
	S L L	259.81	14.00	14.54	0.12	373.30	1.58	52.80	1.96	511.61	4.86	158.18	13.10
	M S S	1991.04	4.03	786.36	6.04	2101.21	14.78	2233.48	144.62	2821.29	165.12	3613.18	205.20
	M S M	2801.70	224.69	823.48	1.60	3122.24	132.52	1639.52	38.08	4433.11	297.70	3533.86	187.60
	M S L	5877.93	479.95	1018.82	0.86	7226.29	373.94	2334.34	23.30	7983.29	893.95	4366.08	183.02
	M M S	1740.77	46.79	450.00	1.90	1948.90	155.90	975.32	73.50	2622.98	127.69	1810.78	191.84
	M M M	1985.03	44.75	356.90	2.60	2406.01	66.00	857.98	37.54	3398.17	315.33	1830.10	185.00
	M M L	2574.42	64.03	423.12	2.16	3722.47	64.14	890.00	25.84	5143.86	610.83	2142.02	182.26
	M L S	1579.89	70.89	205.28	2.02	1773.26	40.52	400.34	32.14	2159.71	75.08	997.70	177.08
	M L M	1706.89	44.84	169.08	2.10	2083.99	93.76	402.56	22.80	2423.10	21.96	974.34	147.96
	M L L	1990.08	98.82	175.00	0.94	2496.55	132.17	394.60	17.58	2872.08	23.17	961.12	124.22
	L S S	4038.96	138.07	1659.60	10.30	4665.85	346.50	3544.64	182.50	6405.06	335.71	7965.86	246.62
	L S M	6024.88	223.95	1761.12	1.62	7095.16	541.33	3744.02	69.26	9806.55	720.66	7441.12	202.00
	L S L	8386.70	310.61	1567.76	1.24	14268.32	181.70	4641.42	34.16	17555.83	1741.95	8878.66	189.24
	L M S	4057.30	87.29	1016.68	7.22	4086.49	102.05	2073.38	128.26	5356.91	260.83	3459.20	211.42
	L M M	4097.24	97.78	654.02	4.14	5224.04	156.20	1810.94	66.54	6968.50	421.69	3405.30	195.22
	L M L	6588.28	82.42	915.48	0.76	6863.49	319.80	1645.66	29.62	10534.39	1082.81	3842.90	188.62
	L L S	3392.76	61.00	360.96	3.88	4002.69	94.77	926.96	61.34	5417.87	289.39	1692.64	193.14
	L L M	3766.66	24.20	368.20	3.66	4601.59	86.73	888.64	45.68	6074.01	284.16	1685.36	188.60
	L L L	4227.36	161.08	364.06	1.10	5713.25	137.12	906.82	31.14	7460.77	702.98	1768.10	184.92

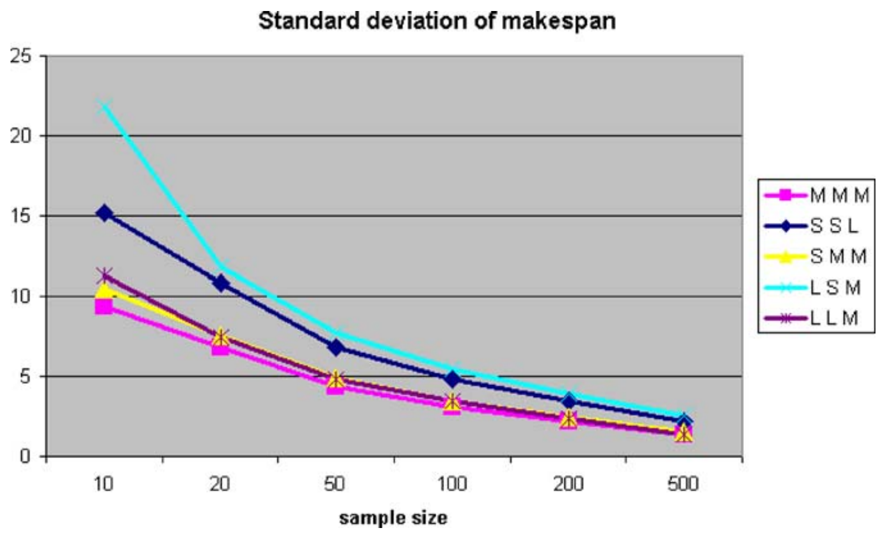

Fig. 4 Standard deviation of makespan

random variables characteristics, as mentioned above, but also with respect to the number m of machines and/or the number n of jobs.

7. Conclusions and further research

In this paper, we consider a general dynamic procedure that addresses any stochastic open-shop scheduling problem subject to random breakdowns as a sequence of stochastic problems without breakdowns. Taking into account that, at any time, the algorithm provides good solutions by considering the available information up to that time, we can say that the final solution is an appropriate and interesting schedule. The procedure solves stochastic open-shop problems by considering the minimization of the expected makespan. An illustrative example and computational experience are also reported. Further research could be directed to apply this approach to general stochastic job-shop scheduling problems subject to random breakdowns. Other research could be focused on bicriteria and multicriteria stochastic open-shop scheduling problems. Also, it would be interesting to study stochastic open-shop scheduling problems where precedence relations among jobs are known.

Acknowledgments This work is partially supported by Spanish Government Research Projects DPI2001-2715-C02-02 and MTM2004-7550 from National Plan of Scientific Research and Technological Development and Innovation, which are co-financed by European Funds of Regional Development.

References

Alcaide D, Rodriguez-Gonzalez A, Sicilia J (2002) An approach to solve the minimum expected makespan flow-shop problem subject to breakdowns. Europ J Oper Res 140(2):384–398

Alcaide D, Sicilia J, Vigo D (1997) A tabu search algorithm for the open shop problem. Top 5(2):283–296

Allahverdi A (1994) Optimal policies for a deterministic two-machine semi-ordered flowshop with maximum lateness objective. Int Trans Oper Res 6:47–54

Allahverdi A (1995) Two-stage production scheduling with separate setup times and stochastic breakdowns. J Oper Res Soc 46:896–904

Allahverdi A (1996) Two-machine proportionate flowshop scheduling with breakdowns to minimize maximum lateness. Comp Oper Res 23:909–916

Allahverdi A (1997) Scheduling in stochastic flowshops with independent setup, processing, and removal times. Comp Oper Res 24:955–960

Allahverdi A (1999) Stochastically minimizing total flowtime in flowshops with no waiting space. Europ J Oper Res 113:101–112

Allahverdi A, Mittenthal J (1994a) Two-machine ordered flowshop scheduling under random breakdowns. Math Comp Model 20:9–17

Allahverdi A, Mittenthal J (1994b) Scheduling on m parallel machines subject to random breakdowns to minimize expected mean flowtime. Naval Res Logist 41:677–682

Allahverdi A, Mittenthal J (1995) Scheduling on a two-machine flowshop subject to random breakdowns with a makespan objective function. Europ J Oper Res 81:376–387

Allahverdi A, Mittenthal J (1998) Dual criteria scheduling on a two-machine flowshop subject to random breakdowns. Int Trans Oper Res 5:317–324

Allahverdi A, Savsar M (2001) Stochastic proportionate flowshop scheduling with setups. Comp Ind Eng 39:357–369

Allahverdi A, Tatari MF (1997) Stochastic machine dominance in flowshops. Comp Ind Eng 32:735–741

Anderson EJ, Glass CA, Potts CN (1997) Machine scheduling. In: Aarts E, Lenstra JK (eds) Local search in combinatorial optimization, Ch. 11. Wiley, New York

Baker KR (1974) Introduction to sequencing and scheduling. Wiley, New York

Blazewicz J, Ecker K, Pesch E, Schmidt G, Weglarz J (2001) Scheduling computer and manufacturing processes. Springer-Verlag, New York

Brucker P (2001) Scheduling algorithms. Springer, Berlin

Cai X, Zhou S (1999) Stochastic scheduling on parallel machines subject to random breakdowns to minimize expected costs for earliness and tardy jobs. Oper Res 47(3):422–437

Fabrycky WJ, Ghare PM, Torgersen PE (1972) Indus oper res. Prentice-Hall, New Jersey

French S (1982) Sequencing and scheduling. An Introduction to the mathematics of the job shop, Wiley, New York

Gonzalez T, Sahni S (1976) Open shop scheduling to minimize finish time. J Assoc Comp Mach 23:665–679

Gonzalez T, Sahni S (1978) Flowshop and jobshop schedules: Complexity and approximation. Oper Res 26(1):665–679

Graham RL, Lawler EL, Lenstra JK, Rinnooy Kan AHG (1979) Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann Discr Math 5:287–326

Grant EL, Leavenworth RS (1996) Statistical quality control. McGraw-Hill, New York

Lawler EL, Lenstra JK, Rinnooy-Kan AHG (1981) Minimizing maximum lateness in a two machine open shop. Math of Oper res 6:153–158. Erratum. Math Oper Res 7:635

Lawler EL, Lenstra JK, Rinnooy-Kan AHG (1982) Recent developments in deterministic sequencing and scheduling: A survey. In: Dempster MAH, Lenstra JK, Rinnooy Kan AHG (eds) Deterministic and stochastic scheduling, NATO Advanced Study Institute Series. Serie C. Reidel Publishing Company, Dordrecht, Holland

Lawler EL, Lenstra JK, Rinnooy Kan AHG, Shmoys DB (1993) Sequencing and scheduling: Algorithms and complexity, In: Graves SC, Rinnooy Kan AHG, Zipkin PH (eds) Handbooks in operations research and management science, vol. 4, Logistics of Production and Inventory, Ch. 9. North Holland, Amsterdam, The Netherlands

Li W, Cao J (1995) Stochastic scheduling on a single machine subject to multiple breakdowns according to different probabilities. Oper Res Lett 18:81–91

Pinedo ML (1981) A note on the two-machine job shop with exponential processing time. Nav Res Logist Quart 28:693–696

Pinedo M (2002) Scheduling: Theory, algorithms and systems. Prentice Hall, Englewood Cliffs, NJ

Pinedo ML, Rammouz E (1988) A note on stochastic scheduling on a single machine subject to breakdowns and repair. Probab Eng Inform Sci 2:41–49

Pinedo ML, Ross SM (1982) Minimizing expected makespan in stochastic open shops. Adv Appl Probab 14:898–911

Pinedo ML, Schrage L (1982) Stochastic shop scheduling: A survey. In: Dempster MAH, Lenstra JK, Rinnooy Kan AHG (eds) Deterministic and stochastic scheduling, NATO Advanced Study Institute Series. Serie C, Reidel Publishing Company, Dordrecht, Holland

Pinedo ML, Wie SH (1986) Inequalities for stochastic flow shops and job shops. Appl Stoch Mod Data Anal 2:61–69

Weiss G (1982) Multiserver stochastic scheduling. In: Dempster MAH, Lenstra JK Rinnooy Kan AHG, (eds), Deterministic and stochastic scheduling, NATO Advanced Study Institute Series. Serie C, Reidel Publishing Company, Dordrecht, Holland

Weiss G (1995) A tutorial in stochastic scheduling. In: P. Chretienne, E.G. Coffman, J.K. Lenstra, Z. Liu (eds) Scheduling theory and its applications. Wiley, New York