A study of maintenance contribution to joint production and preventive maintenance scheduling problems in the robustness framework.

Fatima Benbouzid-Sitayeb, Ahcène Bendjoudi, Samir Benkhellat, Christophe Varnier, Noureddine Zerhouni

A Study of Maintenance Contribution to a Joint Production and Preventive maintenance Scheduling Problem in the Robustness Framework

Fatima Benbouzid-Sitayeb*, Ahcène Bendjoudi and Samir Benkhallat
Laboratoire des Méthodes de Conception de Systèmes (LMCS)
Ecole nationale Supérieure d'Informatique (ESI ex INI)
BP 68M Oued Smar
Algiers 16270, Algeria
E-mail: f_sitayeb@esi.dz
E-mail: fatima_benbouzid@yahoo.fr
E-mail: ahcene.bendjoudi@gmail.com
E-mail: benkhellat_samir@hotmail.com
*Corresponding author

Christophe Varnier and Noureddine Zerhouni
FEMTO-ST Institute, Automatic Control and Micro-Mechatronic Systems Department
UMR CNRS 6174-UFC/ENSMM/UTBM
25, rue Alain Savary
Besançon 25000, France
E-mail: christophe.varnier@ens2m.fr
E-mail: zerhouni@ens2m.fr

Abstract: In this paper, we deal with a joint production and Preventive Maintenance (PM) scheduling problem in the robustness framework. The contributions of this paper are twofold. First, we will establish that the insertion of maintenance activities during production scheduling can hedge against some changes in the shop environment. Furthermore, we will check if respecting the optimal intervals of maintenance activities guarantees a minimal robustness threshold. Then, we will try to identify from the used optimisation criteria those that allow making predictive schedules more robust. The computational experiments in a flowshop show that joint production and PM schedules are more robust than production schedules and maintenance provides an acceptable tradeoff between equipment reliability and performance loss under disruption

Key words: Production, Preventive maintenance, Flowshop, Joint scheduling, Disruption, robustness, Optimization criteria.

1. Introduction

Production scheduling is one of the most important tasks carried out in manufacturing systems. It is responsible for the scheduling of jobs in machines and the specification of the sequence and time to be carried out of operation. Some productive systems present a special configuration that has been widely studied in the literature. This configuration implies a natural ordering of the machines in the shop in that the jobs go through the same machines in the same order. This type of configuration is called 'flowshop'.

In the research literature, production scheduling is usually seen as a function of perfect inputs. The set of orders, capacities of machines, duration of activities and other characteristics of the scheduling problem are assumed to be known and static. Several techniques have been proposed to generate (for a given problem) a unique schedule satisfying shop constraints and providing optimal or near-optimal performance. However, when this precomputed or predictive schedule is released for execution, continual adaptations are required to take uncertainties into account. These uncertainties are related, for example, to machine breakdowns, staffing problems, the unexpected arrival of new orders, the early or late arrival of raw materials and uncertainties in the duration of processing times (Aloulou and Portmann, 2003).

As a result, over the past 20–30 years, one of the most interesting branches of combinatorial optimisation that emerged is robust optimisation. Two new research lines within the operations research community have been initiated with an increasing interest in the use of worst-case

optimisation models: disruption management and robust scheduling. In this paper, we deal with the second research line.

Another task closely related to production scheduling in industrial settings is Preventive Maintenance (PM), understood as the operations or techniques that allow the maintenance or restoration of equipment to a specific state and the guarantee of a given service. These activities conflict since PM activities consume potential production time, but delaying PM because of production demands may increase the probability of machine failure.

Usually, scheduling PM operations and production sequencing are dealt with separately in the literature and, therefore, also in the industry. Therefore, little work has been carried out in which PM scheduling and flowshop scheduling are jointly considered (Lee and Chen, 2000; Cassady and Kutanoglu, 2003; Benbouzid *et al.*, 2003; Aggoune, 2004; Allaoui and Artiba, 2004; Ruiz *et al.*, 2007). There is one account in the literature regarding two joint scheduling strategies (Lee and Chen, 2000) aiming to solve the conflicts between production and maintenance: the sequential strategy that consists of scheduling production jobs, then inserting maintenance tasks, taking production scheduling as a strong constraint, and the integrated strategy that consists of simultaneously scheduling both maintenance and production activities based on their common representation.

In this paper, we develop a model that studies the maintenance contribution to joint production and PM scheduling robustness. In the first step, we build a set of schedules (Benbouzid *et al.*, 2003) restricted to follow a disruption protocol, which allows the comparison of the performances of disrupted joint production and PM schedules to disrupted production ones (without maintenance). Then, we will try to identify from the used optimisation criteria the ones that allow making predictive schedules more robust. The purpose of such a model is to describe the benefits of inserting maintenance activities during production scheduling that can guarantee a minimal robustness threshold.

The rest of the paper is organised as follows. In Section 2, we present a short state-of-the-art concerning both the theoretical and practical aspects of scheduling in the robustness framework. In Section 3, we describe the context of the study: production and maintenance data and the common objective function to optimise. Section 4 introduces our experimental environment through the optimisation criteria used in the tests, then the disruption protocol and disruption algorithm. Section 5 is devoted to the results. Finally, we conclude with some development prospects and extensions to our work.

2. Literature review

The theory of robustness is a relatively new and fast-developing area of combinatorial optimisation. It deals with the uncertainty of problem parameters. The presence of such parameters in optimisation models is caused by the inaccuracy of initial data, the inadequacy of models to real processes, the errors of numerical methods, rounding-off errors and some other factors. Thus, it appears important to identify the model classes in which small changes on the input data lead to small changes in the results, under the worst possible scenario of problem parameters distribution. During the past 10–20 years, many authors concentrated their work on robust optimisation and related approaches in which one optimises against the worst instances that might arise by using the min-max (or some other) objective.

2.1. Robustness in scheduling

In scheduling, it might be important for the disrupted schedule to remain very close to the original one when the release dates are slightly changed and the problem is reoptimised. Two types of robustness are generally mentioned in the literature: *quality robustness* and *solution robustness* (Sörensen, 2001). A solution is called ‘quality robust’ if the quality of this solution is relatively insensitive to changes in the problem data. This type of robustness can also be called ‘robustness in the objective function space’. A solution is called ‘robust’ if it remains approximately the same when changes occur in the input data.
Many scheduling and rescheduling approaches have been proposed in the literature to take into account the presence of uncertainties (disruptions) in the shop floor (Aloulou and Portmann, 2003).

Their aim is to produce schedules that easy recovery or remain feasible after disruption; see, for example, Davenport and Beck (2000), Herroelen and Leus 2005 and Jensen 2001.
These approaches can be classified into four categories: completely reactive scheduling, predictive-reactive scheduling, proactive scheduling and proactive-reactive scheduling. Completely reactive approaches are based on up-to-date information regarding the state of the system (Davenport and Beck, 2000). No predictive schedule is given to the shop floor and decisions are made locally in real time by using priority-dispatching rules. In predictive-reactive approaches, a schedule is generated without considering the possible disruptions. Then, a reactive algorithm is used to maintain the feasibility of the schedule and/or improve its performances (Vieira et al., 2003). Proactive or robust scheduling takes into account the possible disruptions while constructing the original predictive schedule. This allows making the predictive schedule more robust. A robust schedule is defined by Leon et al. (1994) as "a schedule that is insensitive to unforeseen shop floor disruptions given an assumed control policy". This control policy is generally simple. Robust scheduling is appropriate only if, while generating the predictive schedule, uncertainty is known or at least some suspicions about the future are given, thanks to the experience of the decision maker. If uncertainty is completely unknown, a reactive scheduling approach is more suitable.
Most of the work related to robustness in the scheduling theory is based on two important factors: the uncertainties taken into account and the used robustness measures. Uncertainties in scheduling may arise from many sources: machine breakdowns, early or late arrival times, uncertainty of processing times, changes in release data, natural and human factors, *etc*. The most studied uncertainties are related to machine breakdowns (Mignon *et al*., 1995; Sanmarti *et al*., 1995; Lawrence and Sewell, 1997; Balasubramanian and Grossmann, 2002; Alcaide *et al*., 2002). Nevertheless, some work studying the uncertainty of processing times can be found in Mignon *et al.* (1995) and Balasubramanian and Grossmann (2002).
For robustness measures, authors generally follow one of two classes of experiments: the first uses only one robustness measure and several resolution methods (rescheduling approaches are used), whereas the second class uses one or few resolution methods but relate to several robustness measures. In both classes, the robustness measure is known.
No one can guarantee the reliability of input data in the contemporary, dynamic, permanently changing world. This is one reason why almost all modern scheduling techniques try to find a solution (being probably near-optimal, but as close as possible to an optimal one) that is flexible to input data changes.

2.2. Robustness and preventive maintenance

In the preceding section, a brief description of the literature concerning both the theoretical and practical aspects of scheduling in the robustness framework was given. Even though production scheduling and PM planning have received considerable attention from manufacturing industries, in practice, these are made independently despite the relationship between them. Therefore, 'preventive maintenance' does not appear in the literature review presented above. All the studied works focus only on production scheduling.
PM apparently has a negative impact on the performance of a production system. Given that maintenance affects the available production time and the elapsed production time affects the probability of machine failure, this interdependency seems to be overlooked in the literature. Solving production scheduling and PM planning problems independently ignores these inherent conflicts. Our contention is that manufacturing system productivity would benefit from the integration of these decisions. The goal is to reach a compromise between system performance and equipment reliability.
One can measure system reliability without maintenance, knowing that, on the one hand, corrective action never brings the system back to its nominal state and, on the other hand, lots of corrective actions decrease system performance. One can also evaluate the maintenance contribution in the global performance of the system under disruption, which deals with the robustness of the production system. The first proposal has been widely studied in the literature. Some authors studied the impact of maintenance policies on the industrial process (Sherwin, 2000; Vineyard *et al.*, 2000). Others were interested in industrial cases (Sanmarti *et al.*, 1995; Ashayeri *et al*., 1996; Alcaide *et al.*, 2002). In this paper, we will investigate the first proposal.

3. Models and problem statement

In the following subsections, we will first present the production and maintenance data, then the objective functions to optimise.

3.1. Formulation of the static permutation flowshop scheduling problem

One of the most frequent production scheduling problems is the Flowshop Problem (FSP), which represents a lot of industrial cases. The FSP can be stated as follows. Each n job 1… n has to be processed on m machines 1… m, in that order. The processing time of job i on machine j is p_{ij}. The processing times are fixed, non-negative and may be 0 if a job is not processed on some machines. Further assumptions are that each job can be processed on only one machine at a time, the operations are not preemptable, the jobs are available for processing at time 0 and set-up times are sequence-independent. Here, we consider the Permutation Flowshop Problem (PFSP), *i.e.*, the same job order is chosen on every machine. The objective is to find a sequence, *i.e.*, a permutation of the numbers 1…n that minimises the completion time C_{max} (makespan) of the last job. In this case, the problem is denoted by $F_m/ /C_{max}$ following Reeves' (1995) notation. The problem is NP-hard (Garey *et al.*, 1976); only some special cases can be solved efficiently (Johnson, 1954).

3.2. Preventive maintenance and scheduling

Maintenance is understood as any activity carried out on a system to maintain or restore it to a specific state (Biroloni, 2004). Maintenance operations can be classified into two large groups: Corrective Maintenance (CM) and PM. CM is carried out when the failure has already taken place. PM consists of carrying out operations in machines and equipment before the failure or breakdown takes place and at previously established time intervals. The objective of PM is to prevent failures and, therefore, to minimise the probability of failure. The advantage of PM is that the system is always in good condition, thus reducing the risk of unexpected failures.

The choice of PM for this study is a consequence of its planned aspect that makes it the most adapted for maintenance scheduling. In this case, the search for a production schedule will be correlated to the search for maintenance planning, which is predefined and easy to implement.

PM can be stated as follows. Each machine is maintained periodically at known time intervals. The maintenance tasks are periodic interventions occurring every T periods and each occurrence depends on the one preceding it on the same machine. A maintenance task consists of elementary operations wherein processing time p' is evaluated with more or less certainty. Moreover, the periodicity T of these tasks can vary in a tolerance interval noted $[T_{min}, T_{max}]$. This interval gives some flexibility to plan maintenance tasks while respecting the production constraint, disturbing the least possible production schedule and respecting maintenance equipment periodicity.

The tolerance interval represents a compromise between the maintenance cost and machine unavailability. If the period is lower than T_{min} (zone 1 in Figure 1), maintenance interventions will be too frequent compared to the machine's real need and, thus, induce a too high maintenance cost. Moreover, if some maintenance tasks are programmed after T_{max} (zone 2 in Figure 1), breakdowns and CM actions are likely to appear. As a result, the machine would be unavailable. This established fact induces a maintenance cost increase and performance loss. However, an intervention planned in interval $[T_{min}, T_{max}]$ will induce a relatively constant maintenance cost.

Machine j can be subject to several different maintenance tasks which will be repeated periodically. Let:

- $\mathbf{M_{ij}}$: the maintenance task i on the machine j.
- $\mathbf{T_{ij}}$**:** periodicity of the maintenance task M_{ij}.
- $\mathbf{Tmin_{ij}}$: earliest time separating two consecutive occurrences of M_{ij};
- $\mathbf{Tmax_{ij}}$: latest time separating two consecutive occurrences of M_{ij};
- $\mathbf{p'_{ij}}$: processing time of task M_{ij}. It is supposed to be known and constant.

Ideally, a maintenance task is programmed in interval $[T_{min}, T_{max}]$. However, it can be programmed before T_{min} and considered advance (this advance is noted E') or programmed after T_{max} and considered late (this delay is noted L'). The earliness and tardiness of the *k*-th occurrence of maintenance task M_{ij} are computed as follows:

- **t'_{ijk} =** execution time of the k^{th} occurrence of the maintenance task M_{ij}.
- **E'_{ijk}** = earliness execution time of the k^{th} occurrence of the maintenance task M_{ij}.

$$E'_{ijk} = \max (0, t'_{ijk}+p'_{ij}+Tmin_{ij} - t'_{ijk-1})$$

- **L'_{ijk}**: Tardiness execution time of the k^{th} occurrence of the maintenance task M_{ij}.

$$L'_{ijk} = \max (0, t'_{ijk-1} - t'_{ijk} - p'_{ij} - Tmax_{ij})$$

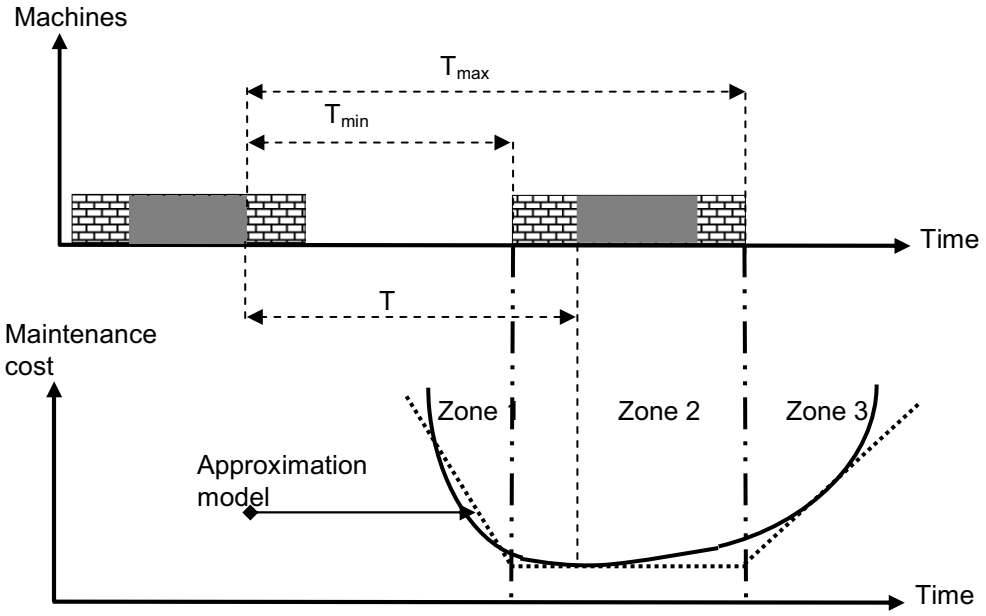


Figure 1: Tolerance interval of a maintenance task

3.3. Objective functions

The goal of joint scheduling is to propose a method that provides common planning for production jobs and maintenance tasks. Thus, the objective of optimisation is to find a compromise between the target objective maintenance and production functions.

The constraints imposed by the customers to their suppliers are often expressed in terms of time, which naturally lead us to the minimisation of the makespan. One will note f_1, the production objective function:

$$f_1 = C_{\max} = Max(c_{ij})^{1} \qquad [1]$$

We introduced in the preceding section a cost curve that depends on the period wherein maintenance activities are done (time). This curve can be established on the base of a mathematical model. In this study, we investigate the robustness of an integrated production scheduling and PM planning model considering that the respective maintenance activities' periods can be hedged against some machine failure. The purpose of this work is to optimise time, not cost. From the supplier point of view, the respective maintenance periods influence the constraints of the production system. One will note f_2, the maintenance objective function:

$$f_2 = \sum_{j=1}^{m} \sum_{k=1}^{Maxj} E'_{jk} + L'_{jk}{}^{2} \qquad [2]$$

To optimise the two criteria, we take into account the following common global objective function:

$$f = \alpha f_1 + \beta f_2 \qquad [3]$$

---

[1] c_{ij} is the completation time of task i on the machine j

[2] Maxj represents the effective occurrence number of the maintenance task M_j.

(α,β) are weights that will measure the respective contributions of production and maintenance in the global objective function. The goal is not to study multicriteria optimisation, but only to measure the impact of production or maintenance on the global objective function. For that, parameters α and β can depend on the number of tasks, processing time or problem size. They are independent and noncomplementary.

4. Experimental environment

The aim of this study is to evaluate the maintenance contribution to the robustness of joint production and maintenance scheduling. For that, we will try to answer the following questions:

1. Does maintenance contribute to scheduling robustness?

The answer to this question will determine if maintenance contributes to joint scheduling robustness. In static environments, the insertion of maintenance tasks in the schedule decrease some optimisation criteria such as makespan because of the
delay of some production jobs. But in dynamic environments, where disruptions can occur, it can improve it by optimising machine reliability (a recently maintained machine will have a weak probability of performance degradation).
In this paper, we do not propose robust joint scheduling, but rather to register the generation of joint production and PM schedules as a robust approach. Our goal is to establish that the performance decrease of joint production and PM scheduling after the insertion of maintenance activities is lower than the decrease of production scheduling performance under disruptions.

2. Do some optimization criteria have any influence on scheduling robustness?

The answer to this question implies the study of the influence of some classical optimisation criteria on scheduling robustness. This question is very important if robust schedules exist, but we can show that neither the maintenance nor scheduling methods influences this robustness. At this time, the goal consists of searching the optimisation criteria under this improvement.

In the following, we will first present the selected optimization criteria for this study. Then, we will introduce our disruption protocol and the disruption algorithm;

4.1. Optimization criteria

To answer the second question of this study, we chose the following criteria in the production objective function (Section 3.3) (Pinedo, 1995): C_{max}, flow time, idle time, $\sum Ti$ and, for maintenance, the sum of the advances and delays in maintenance (Section 3.3). The global objective function takes into consideration the production optimisation criteria and the maintenance one.
The criteria to be optimised in the production objective function were chosen because of the bound existing between them. Of course, if C_{max} varies, the others vary in the same direction.

4.2. Disruption protocol

The shop environment is subject to disruptions related to production jobs. We model a disruption as an increase of processing times according to levels and classes. This classification refers to the study of Lawrence and Sewell (1997).
In the following subsections, we will first define disruption levels and classes and the concept of overlapping, which is a consequence of tasks disruption. Then, the maintenance effect when disruption occurs is introduced. It is modelled as reliability and disruption intervals.

4.2.1. Definition of disruption levels

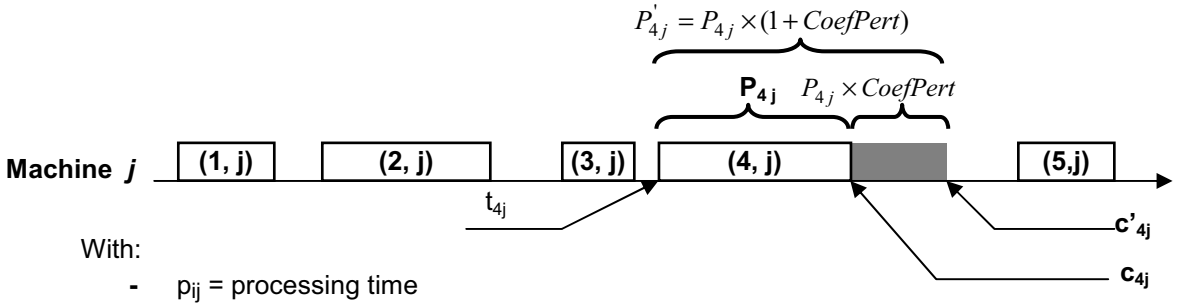
A disruption level determines the production job's processing time increase. Ten levels of disruption in processing times were studied. They represent the intervals of possible processing time increases. At each level, the maximum increase of the disturbed task processing time is 10% of the original processing time. This variation is obtained by a coefficient noted $CoefPert_{ij}$ for a task (i,j). This coefficient is generated by a uniform law U.
Table 1 defines the disruptions levels and the uniform variable associate with each level.

Level	Interval	Variation
1	] 0%, 10%]	$CoefPert_1$ ➔ U [0, 0.1]
2	] 10%, 20%]	$CoefPert_2$ ➔ U [0.1, 0.2]
.	.	
.	.	
.	.	
9	] 80%, 90%]	$CoefPert_9$ ➔ U [0.8, 0.9]
10	] 90%, 100%]	$CoefPert_{10}$ ➔ U [0.9, 1]

Table 1: Disruption levels

Figure 2 illustrates the new processing time (P'_i) of a disturbed job defined as:

$$P'_i = P_i \times (1 + CoefPert_i)$$



With:

- p_{ij} = processing time
- p'_{ij} = new processing time after disruption
- t_{ij} : starting time
- c_{ij} : completion time.
- $CoefPert_i$: disruption coefficient of the task i.

Figure 2: Processing time of a disturbed job

4.2.2. Definition of disruption classes

A *disruption class* represents the number of tasks to disturb in the schedule. We define three disruptions classes: weak, average and strong. Table 2 gives the number of disturbed tasks in the schedule for each class.

Class	Interval of variation	Number of disturbed tasks
Weak	[0%, 40%]	u ➔ U [0, 0.4] / n x u disturbed tasks
Average	] 40%, 70%]	u ➔ U] 0.4, 0.7] / n x u disturbed tasks
strong	] 70%, 100%]	u ➔ U] 0.7, 1] / n x u disturbed tasks

n: number of tasks

Table 2: Disruption classes

4.2.3. Overlapping

The increase of production tasks' processing times can cause overlapping between two or several tasks. An overlap between two tasks is defined as "the interlacing or the superposition of two tasks after the right-shifting of the first one for an unspecified reason". Overlapping can occur between two production jobs or between a production job and a maintenance task. Overlapping can be classified into two large groups (Figure 3): *horizontal overlapping*, which occurs between two successive tasks on the same machine, and *vertical overlapping*, which occurs on the same task executed on two successive machines.

To solve overlapping, right-shifting rescheduling is applied because it preserves the tasks' execution order. Nevertheless, in the case of joint production and PM scheduling, we will check if the maintenance task is still in its tolerance interval after right shifting. Otherwise, permutations between

the production job that caused the overlap and the maintenance one will be done to put it back in its tolerance interval.

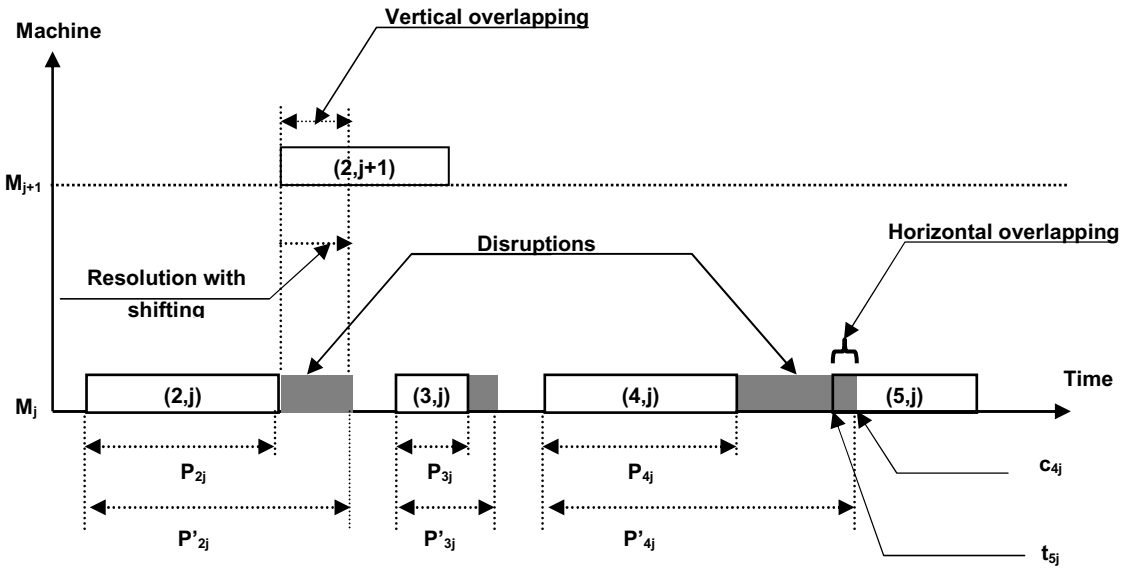


Figure 3: Horizontal and Vertical overlapping between two tasks

4.2.4. Reliability and disruption intervals

After machine maintenance, its reliability will increase and can hedge against several disruptions. We model the maintenance effect on machine reliability by two intervals: a reliability interval (after a PM action on the machine) and a disruption interval that succeeds the reliability one (Figure 4).
In the reliability interval, no disruption on the production job's processing time can occur because the machine has been recently maintained and is, therefore, still reliable. This interval is considered maintenance data; it depends only on the machine concerned with maintenance. In the disruption interval, disruptions can occur progressively.

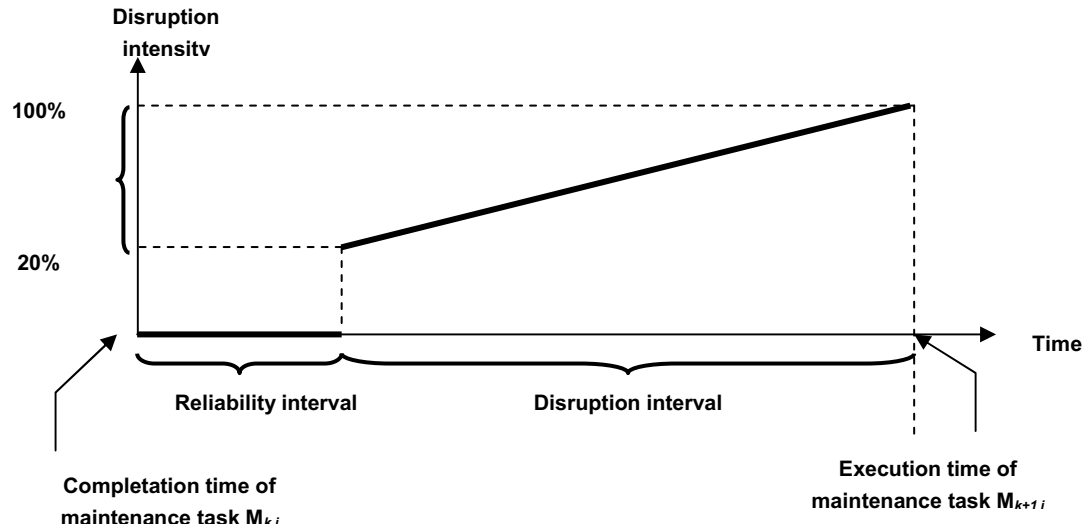


Figure 4: Reliability and disruption intervals

4.3. Disruption algorithm

The four steps (generation, disruption, classification and validation) of the algorithm that we propose to study the maintenance contribution to robustness of joint production and PM scheduling under disruption are outlined below.

Step 1: Generation

We build three sets of near-optimal schedules with a Genetic Algorithm (GA) by following the procedure given by Benbouzid *et al.* (2003). The first set is composed only of production schedules (PGA) resulting from the first step of the sequential strategy. The second one is composed of the joint production and PM schedules resulting from the second step of the sequential strategy (SGA) (after the insertion of maintenance). The last one is also composed of the joint production and PM schedules resulting from the integrated strategy (IGA). The aim of this distinction between the generation strategies of joint scheduling is to see whether the generation method has any influence on schedules' robustness.

Step 2: Disruption

A schedule can be disturbed at three different levels, according to the three sets: at the generation of the production schedule (the first step of joint production and PM scheduling generation with the sequential strategy), after the insertion of maintenance tasks (the second step of the sequential strategy) and after the generation of the schedule in the case of the integrated strategy. The schedules of each set are first optimised with the optimisation criteria selected for the study (Section 4.1). As a result, 12 (three sets and four optimisation criteria) primary sets are obtained. Then, all the schedules are disrupted according to the disruption classes and levels defined in Section 4.2. Finally, $12 \cdot 3 \cdot 10$ sets of disturbed schedules are obtained (12 primary sets disturbed according to the 10 disruption levels and 3 disruption classes). At the end of each disruption process, we compute the differences between the performances of the original schedule and the disturbed one.

In the majority of the experiments in the robustness framework, the loss allowed after disruption is 50% of the original performance. To be able to study the disturbed schedules, we subdivided the loss of performance after the disruptions in intervals and affixed a label to each one. We set 50% as a maximum increase in the objective function. Beyond this value, generating a new schedule will be better. Each interval represents the increase rate of the disturbed schedule performance criterion. The labels and the corresponding performance loss intervals are illustrated in Table 3.

Performance loss	Label
[0, 5%]	Very Good (VG)
] 5%, 15%]	Good (G)
] 15%, 35%]	Average (A)
] 35%, 50%]	Bad (B)
> 50%	Very Bad (VB)

Table 3: labels and corresponding intervals

Step 3: Classification

The classification process makes it possible to evaluate the maintenance effect on the robustness of joint scheduling and identify the criteria that are likely to improve schedule robustness. For each set, we establish a schedule classification based on their original performance.

Step 4: Validation

This step aims to confirm the results of the preceding step. The validation process is applied only when some criteria emerge from the classification process. This validation will be done as follows: we build sets of schedules that will be optimised according to the robust criteria deduced from the validation step. If we obtain the same criteria after the disruption and classification steps, then we can affirm that these criteria are able to improve robustness.

5. Computational results and discussions

In this section, we present some of the obtained results. The first type of experiments concerns the maintenance effects on robustness of joint production and PM scheduling and their ability to provide to joint production and PM schedules a good tradeoff between reliability and performance under disruption. The experiments show that joint production and PM schedules are more robust than production ones.

In the second type, we test the emergence of performance criteria that can improve robustness under disruption. We show that none of the four tested performance criteria are able to improve scheduling robustness.

In the following subsections, the generation scheme of the studied problems and the disruptions protocol are detailed. The two types of experiments done for this study are then introduced.

5.1. Problem's generation scheme.

The tested problems are generated by following the procedure given by Taillard (1993), Reeves (1995) and Carlier (1978) for permutation flowshop scheduling problems. These problems have different sizes (from 5 to 100 jobs and from 3 to 20 machines), but do not include maintenance data. For those, we developed a generator of random maintenance tasks to generate benchmarks in PM. The used parameters are the number of machines and the maintenance task parameter (T, T_{min} and T_{max}). Each parameter is limited by a minimal and a maximum value to avoid having identical values. To carry out our tests, we generated only one maintenance task per machine for each problem. Moreover, the processing time of a maintenance task is identical for all its occurrences.

For the search of robustness criteria, production objective function f_1 consists of (Pinedo, 1995): makespan (C_{max}), flow time, $\sum Ti$ and idle time. Maintenance objective function f_2 is the minimisation of the sum of the delays and advances in maintenance (Section 3.1). The contributions of the production and maintenance objective functions in the global objective function f are equal to 1 ($\alpha = 1$ and $\beta = 1$). The proposed common-weighted global objective function will allow tackling the problem in a simplified way.

Each schedule was generated with a GA. The results of the GA are obtained after 100 executions of the method. The best result and the associated parameters are saved.

5.2. Problem's disruption scheme.

In this study, we consider production jobs' processing time disruption. We used two types of reliability intervals (Section 3.4.2) that represent a percentage of maintenance task periodicity T: 5 and 25% of T. The disruptions are generated according to levels and classes. We generated 20 schedules per set. For each set, all the schedules were optimised with the four optimisation criteria selected for this study (3 sets · 20 schedules · 4 criteria). Then, the schedules were disturbed according to the defined classes and levels (10 · 3). For each disruption scheme (a specified schedule optimised with a specified criterion, then disrupted according to a class and a level), we did 100 tests and took the average performance into account. To ease the classification step, we qualify as an *acceptable schedule* any schedule whose deviation of its original performance remains in an acceptable interval after experimentation. Its labels are then either Very Good (VG), Good (G) or Average (A). It is *rejected* if the obtained label after experimentation is Bad (B) or Very Bad (VB).

In the following subsections, we present the results of the two experiments carried out.

5.3. Maintenance contribution to joint scheduling robustness

Figure 5 presents the distribution of acceptable cases for the three studied sets: PGA, SGA and IGA.

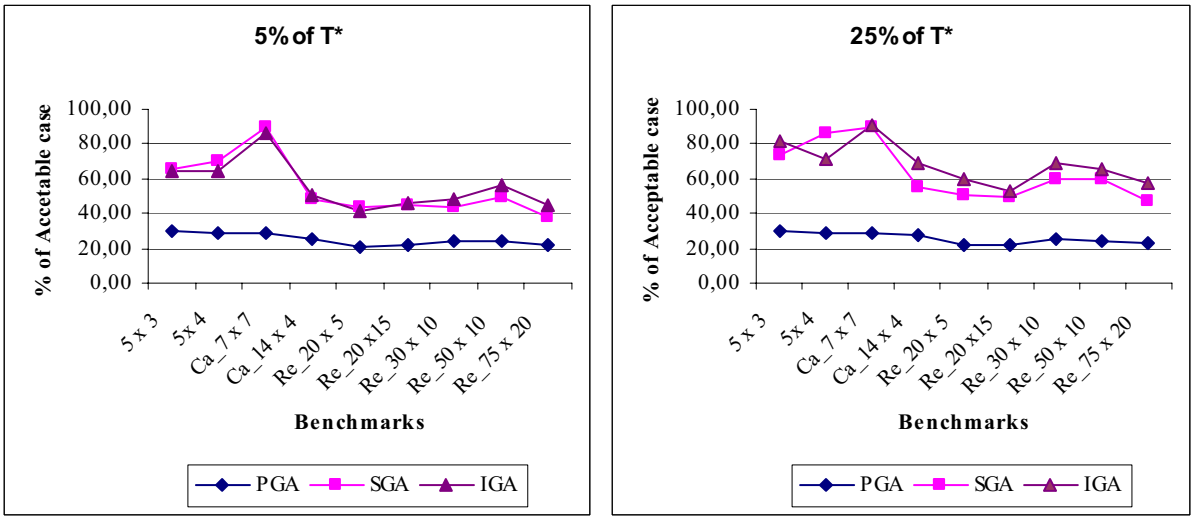


Figure 5: Distribution of acceptable cases for each sets and reliability interval.

We notice that the joint production and PM schedules perform better than the production one. That confirms the maintenance contribution to joint schedules' robustness. For all the benchmarks studied,

we notice that the performance loss of disturbed joint schedules is less than the production ones. Approximately 75% of the disturbed production schedules (all levels and classes were considered) are rejected. Moreover, 25% of the remaining cases have only A or G labels. On the other hand, in the case of joint production and PM schedules, approximately 55% of the cases for SGA and 64% of the cases for IGA are acceptable.
The performance of joint disturbed schedules decreases starting from a high disruption protocol (high disruption class and level) compared to the production disturbed ones. Therefore, there is an increase in the percentage of acceptable joint schedules compared to the production ones.
The results for the second reliability interval (25% of T) are better than the first one (5% of T) because the first interval is larger and so is the period where no disruptions are allowed after maintenance. For the reliability interval 5% of T, the distributions of SGA and IGA sets are almost similar. For the second interval, the IGA set gives the best results for small benchmarks and the SGA set, for large ones.
Finally, it is important to note that the results of the small benchmarks (SGA and IGA) are better than those of large ones. The reason for this good performance is related to the number of machines and tasks. Even for the lowest disruption levels and classes, the large benchmark disruption (lots of tasks) decreases in a significant way. The percentage of average acceptable cases for small benchmarks is 65% against 55% for the largest acceptable cases.
As a conclusion, according to the experiments presented in this part of our study, joint production and PM schedules (SGA and IGA) generate more acceptable cases than production ones (PGA) without considering the benchmark size. The insertion of maintenance tasks in production schedules decreases its performance (while remaining in acceptable limits), but it also brings an unquestionable profit in terms of robustness.
As joint production and PM schedules perform better than production schedules, we will restrict the second experiment to joint production and PM sets.

5.4. Criterion whose optimization makes schedules robust

Figure 6 presents the distribution of the percentage of acceptable cases by performance criterion and reliability interval for SGA sets, regardless of the disruption levels and classes.

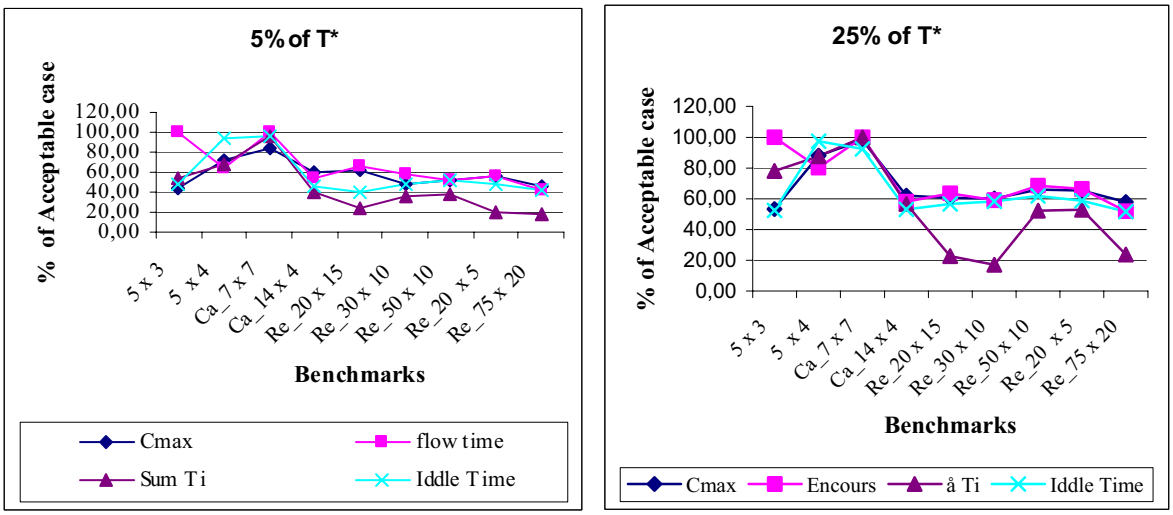


Figure 6: Distribution of acceptable cases per performance criterion and reliability interval for SGA set.

We notice that the flow time and idle time are the criteria that obtain the best results. C_{max} and $\sum Ti$ are the two most unfavourable criteria with regard to robustness.
The disruption process acts on the increase of jobs' processing time. To solve the conflicts that disruptions generate, we use right-shifting rescheduling. It consists of shifting tasks to the right until the conflict generated by disruption is solved. The criteria whose performance worsens more with this technique are C_{max} and $\sum Ti$.
Two points are to be noted:

- The first relates to the reliability interval. On the one hand, we notice an improvement in the percentage of acceptable cases without modifying the general distribution. On the other hand, in the case of the reliability interval of 25% of T, one notes a clear deterioration of $\sum Ti$.

- The second one relates to benchmark size. For small benchmarks, the flow time and idle time are most favourable to robustness. For large benchmarks, the distribution of the four criteria is similar and unfavourable to robustness.

Figure 7 presents the distribution of the percentage of acceptable cases by performance criterion and reliability interval for IGA sets, regardless of disruption levels and classes.

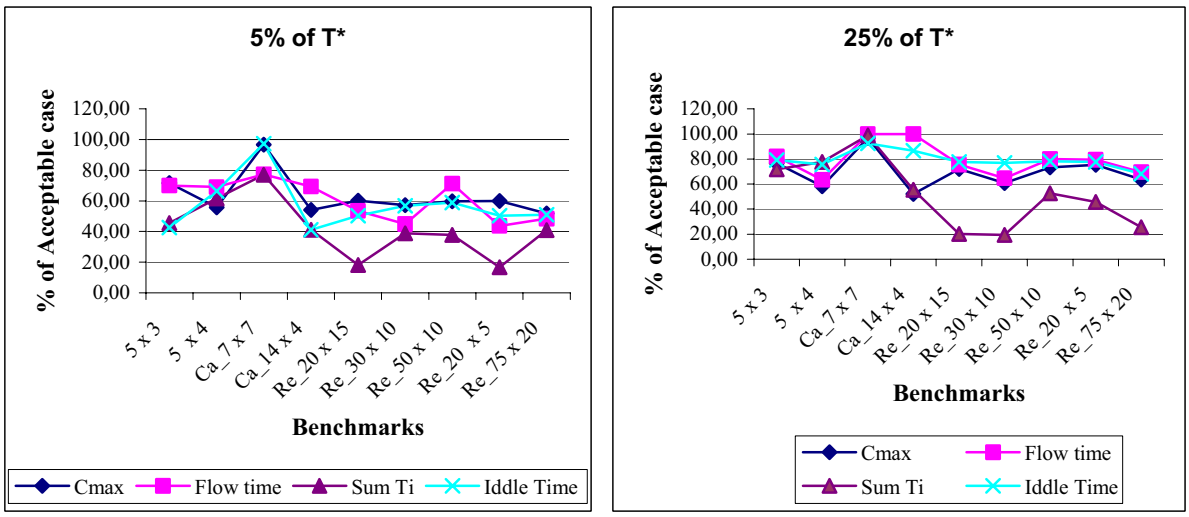


Figure 7: Distribution of acceptable cases per performance criterion and reliability interval for IGA set.

We obtain the same conclusions for the IGA set. Even though the flow time and idle time give the best results, they remain completely dependent on the disruption protocol and benchmark size. It explains the result of 100% acceptable cases for small benchmarks. The robustness criteria emerge only for the first levels of disruption and weak or average classes.
Indeed, there are two tendencies in Figure 7. The first shows that the results of small benchmarks are better than those of large benchmarks. For the second tendency, one notes a clear deterioration of certain criteria (C_{max} and $\sum Ti$).
Finally, the results with a reliability interval of 25% of T are better however it do not have any influence on the general tendency of the distribution.
The flow time and idle time obtain the greatest number of acceptable labels because in the majority of the cases, they have VG and G labels regardless of disruption levels and classes.
The large benchmarks are sensitive to disruptions because the important number of tasks limits schedule flexibility under disruptions. Moreover, as the disruption level and the number of disturbed tasks grow, schedule performance becomes worse. However, if the benchmark is small (few machines and few tasks), the flow time and idle time become important because they represent the flexibility suitable for hedging against some disruptions, making the schedule more robust.
As a conclusion, for the second experiment, there is no robust criterion among the selected ones. We cannot confirm the emergence of a criterion that ensures a robustness threshold for joint production and PM scheduling. There will be no validation process for this classification.
The studied criteria remain completely dependent on the disruption protocol and benchmark sizes. We can only note that the flow time and idle time obtained the best labels for the two studied sets. Also, the results of the IGA set are better than those of the SGA set.
For the reliability interval, the results for interval 25% T are better for the flow time and idle time. But there is a clear deterioration of C_{max} and $\sum Ti$.

6. Conclusion

We considered in this paper the joint production and PM scheduling problem in the permutation flowshop and some optimisation criteria as production objective functions. The shop environment is subject to disruptions related to production jobs' processing time.
We proposed a proactive approach to study the maintenance effect on the robustness of joint production and PM scheduling. We used three different sets of schedules generated with GAs in the

permutation flowshop: a set of production schedules and two sets of joint production and PM schedules. The first was generated by the sequential strategy and the second, by the integrated strategy. We defined disruption classes and levels to study the variation of the schedules' performances under disruptions. Four performance criteria have been used to study their influence on the schedules' robustness.

Several tests were performed to evaluate the maintenance effect on joint production and PM schedules under disruptions and the use of the proposed approach in permutation flowshop conditions.

The first type of experiment concerned the maintenance effects on the robustness of joint production and PM schedules and their ability to provide to a good tradeoff between reliability and performance. The experiments showed that joint production and PM schedules are more robust than production schedules and maintenance allows providing an acceptable trade-off between equipment reliability and performance loss under disruption.

In the second type of experiment, we tested the emergence of performance criteria that can improve robustness under disruption. We showed that none of the four tested performance criteria are able to improve scheduling robustness. However, by making distinctions during simulations between the results obtained by large and small benchmarks, the results showed that for small benchmarks, the flow time and idle time gave the best results.

For this work, several improvements can be made at the level of the implemented methods and at the level of the considered constraints:

- Future researches can study the impact of the weights on the obtained results and how sensitive the results are with regard to the chosen weights. We can investigate a bicriteria optimisation choosing $\alpha + \beta = 1$.
- The results obtained for flowshop scheduling problems gave us some insight to extend our approach to more complex scheduling problems.

Acknowledgements

I would like to thank Dr. Ing Mouloud Koudil and Dr. Ing Karima Benatchchba, associate professors at Ecole nationale Supérieure d'Informatique (ESI ex INI) of Algiers (Algeria), for their contributions and help.

7. References

Aggoune, R. (2004) 'Minimizing the makespan for the flow shop scheduling problem with availability constraints', *European Journal of Operational Research*, Vol. 153, pp.534–543.

Alcaide, D., Rodriguez-Gonzalez, A. and Sicilia, J. (2002) 'An approach to solve the minimum expected makespan flowshop problem subject to breakdowns', *European Journal of Operational Research*, Vol. 140, pp.384–398.

Allaoui, H. and Artiba, A. (2004) 'Integrating simulation and optimization to scheduling a hybrid flow shop with maintenance constraints', *Computers and Industrial Engineering*, Vol. 47, pp.431–450.

Aloulou, M.A. and Portmann, M-C. (2003) 'An efficient proactive-reactive scheduling approach to hedge against shop floor disturbance', *Multidisciplinary International Conference on Scheduling: Theory and Applications – MISTA'2003*, Vol. 1, pp.37–362.

Ashayeri, J., Teelen, A. and Selen, W. (1996) 'A production and maintenance planning model for the process industry', *International Journal of Production Research*, Vol. 34, No. 12, pp.3311–3326.

Balasubramanian, J. and Grossmann, I.E. (2002) 'A novel branch and bound algorithm for scheduling flowshop plants with uncertain processing time', *Computers and Chemical Engineering*, Vol. 26, pp.41–57.

Benbouzid, F., Varnier, C. and Zerhouni, N. (2003) 'Resolution of joint maintenance/production scheduling by sequential and integrated strategies', *IWANN2003 Conference*, 3–6 June, Spain, LNCS, Vol. 2687, pp.782–789.

Biroloni, A. (2004) *Reliability Engineering, Theory and Practice*, 4th ed., Berlin: Springer.

Carlier, J. (1978) 'Ordonnancements a contraintes disjonctives', *R.A.I.R.O. Operations Research*, Vol. 12, pp.333–351.

Cassady, C.R. and Kutanoglu, E. (2003) 'Minimizing job tardiness using integrated preventive maintenance planning and production scheduling', *IIE Transactions*, Vol. 35, No. 6, pp.503–513.

Davenport, A.J. and Beck, J.C. (2000) 'A survey of techniques for scheduling with uncertainty', http:\eil.utoronto.ca/profiles/chris/chris.papers.html.

Garey, M.R., Johnson, D.S. and Sethi, R. (1976) 'The complexity of flowshop and jobshop scheduling', *Mathematics of Operations Researches*, Vol. 1, pp.117–129.

Herroelen, W. and Leus, R. (2005) 'Project scheduling under uncertainty – survey and research potentials',

European Journal of Operational Research, Vol. 165, No. 2, pp.289–306.

Jensen, M.T. (2001) 'Robust and flexible scheduling with evolutionary computation', PhD thesis, University of Aarhus, Department of Computer Science, Denmark.

Johnson, S. (1954) 'Optimal two- and three-stage production scheduling with setup times included', *Naval Research Logistics Quarterly*, Vol. 1, pp.61–68.

Lawrence, S.R. and Sewell, E.C. (1997) 'Heuristic, optimal, static and dynamic schedules when processing times are uncertain', *Journal of Operations Management*, Vol. 15, pp.71–82.

Lee, C.Y. and Chen, Z.L. (2000) 'Scheduling jobs and maintenance activities on parallel machines', *Naval Research Logistics*, Vol. 47, pp.145–165.

Leon, V.J., Wu, S.D. and Storer, R.H. (1994) 'Robustness measures and robust scheduling for job shops', *IIE Transactions*, Vol. 26, No. 5, pp.32–43.

Mignon, D.J., Honkomp, S.J. and Reklaitis, G.V. (1995) 'A framework for investigating schedule robustness under uncertainty', *Computers and Chemical Engineering*, Vol. 21, pp.S615–S620.

Pinedo, M. (1995) *Scheduling: Theory, Algorithms, and Systems*, Series in Industrial and Systems Engineering, Prentice Hall International edition.

Reeves, C.R. (1995) 'A genetic algorithm for flowshop sequencing', *Computer Operational Research*, Vol. 22, pp.5–13.

Ruiz, R., García-Díaz, J.C. and Maroto, C. (2007) 'Considering scheduling and preventive maintenance in the flowshop sequencing problem', *Computers and Operations Research*, Vol. 34, pp.3314–3330.

Sanmarti, E., Espuna, A. and Puigjaner, L. (1995) 'Effects of equipment failure uncertainty in batch production scheduling', *Computer Chemical Engineering*, Vol. 19, pp.S565–S570.

Sherwin, D. (2000) 'A review of overall models for maintenance management', *Journal of Quality in Maintenance Engineering*, Vol. 6, No. 3, pp.138–164.

Sörensen, K. (2001) 'Tabu searching for robust solutions', *Metaheuristics International Conference*, Porto, Portugal, pp.707–712.

Taillard, E. (1993) 'Benchmarks for basic scheduling problems', *European Journal of Operational Research*, Vol. 64, pp.278–285.

Vieira, G.E., Herrmann, J.W. and Lin, E. (2003) 'Rescheduling manufacturing systems: a framework of strategies, policies, and methods', *Journal of Scheduling*, Vol. 6, No. 1, pp.35–58.

Vineyard, M., Amoako-Gyampah, K. and Meredith, J.R. (2000) 'An evaluation of maintenance policies for flexible manufacturing systems', *International Journal of Operations and Production Management*, Vol. 20, No. 4, pp.409–426.