

On the robustness of joint production and maintenance scheduling in presence of uncertainties

Abdelhamid Boudjelida¹ 

Received: 20 February 2015 / Accepted: 30 January 2017 / Published online: 11 February 2017
© Springer Science+Business Media New York 2017

Abstract Production and preventive maintenance are very important functions in industry which act on the same resources. However, in most real workshops, the scheduling of their respective activities is independent and the constraint that they cannot be accomplished at the same time is rarely considered. Therefore, we are facing a joint scheduling problem of production and preventive maintenance tasks. In addition, this joint scheduling risks at any moment to deviate from the theoretical desired performances when facing disturbances due to various causes. Thus, we must still seek the most robust scheduling, i.e. the one that resists to uncertainties. This paper proposes a new approach to study robustness of joint production and maintenance scheduling in permutation flow shop workshops. The studied scheduling are generated according to two strategies: sequential and integrated. As methods of scheduling resolution, we will consider the well-known ants colony optimization, genetic algorithm, tabu search and some hybridizations of these methods. Our approach can be applied to other joint scheduling generating methods. In particular, we study how insertion of maintenance tasks can contribute to the robustness of production scheduling and how some scheduling strategies and methods are more robust than others. Several experimental results show the merits of our approach.

Keywords Production scheduling · Maintenance · Robustness · Uncertainties · Flow shop · Metaheuristics

Introduction

To remain competitive, companies must focus on a continuous improvement of their processes which must meet an increased customers needs of quality, cost and time of availability. This improvement is supported by a quick development of new information and communication technologies. In this context, production scheduling and preventive maintenance are important mechanisms that affect performances of an industrial company. These two functions, generally operate on the same resources which may cause some conflicts between their respective objectives. However, in most real workshops their activities are independents and resources immobilization for maintenance tasks is rarely planned so that it does not interfere with productivity. In addition, cost incurred by a failure (involving a corrective maintenance, unscheduled stop of production, significant delivery delays, etc.) can be much higher than a scheduled stop of production for a preventive maintenance. Therefore, we are facing a joint scheduling problem of production and preventive maintenance tasks, with objectives of respecting, firstly, constraints of delay, cost and quality, and secondly, constraints of functioning security that ensure sustainability of production tools.

In this paper, we are interested in joint production and maintenance scheduling in permutation flow shop workshops (Blazewicz et al. 1996; Malakooti 2013). The complexity studies have shown that usually the flow shop scheduling problems lie in the NP-hard problem class (Rudek and Rudek 2013). Difficulty especially rises as the number of involved jobs or machines increases. In such a situation, despite the spectacular evolutions in computers calculation power, techniques that give the exact solution become insufficient. Thus, several methods were developed to deal with this issue in case of production scheduling (Gen and Lin 2014; Yagmahan and Yenisey 2009; Yenisey and Yagmahan 2014).

✉ Abdelhamid Boudjelida
abdelhamid.boudjelida.1@ulaval.ca

¹ Management Information Systems Department, Laval University, 2325, rue de la Terrasse, Pavillon Palasis-Prince, Quebec, QC, Canada

Among these methods, our study provides an implementation of Ants Colony Optimisation (ACO) (Dorigo 1992), Genetic Algorithms (GA) (Goldberg 1989), Tabu Search (TS) (Glover 1986) and two hybrid GA and ACO approaches. Choice of these methods is not arbitrary but based on the fact that they allow us to obtain solutions as close as possible to the optimal one with a reasonable processing time. In case of only production flow shop scheduling, performances of these methods were demonstrated in works of : Alaykyran et al. (2007), Marimuthu et al. (2009) for Ants Colony Optimisation, Chiou et al. (2012), Ventura and Yoon (2013) for Genetic Algorithms and Bozejko et al. (2013), Grabowski and Wodecki (2004) for Tabu Search. We propose two different strategies to implement these methods for joint production and maintenance scheduling : sequential and integrated. The main difference between these two strategies is the way in which maintenance tasks will be processed during the joint scheduling construction.

The joint scheduling generated by these methods and strategies may deviate, at any time, from the desired theoretical performances. This deviation occurs when the joint scheduling faces disturbances due to various causes (machines failures, operators overloading, project objectives changing, etc.). Therefore, we must still search through all possible joint scheduling, the most robust ones, i.e. the most resisting scheduling to uncertainties which represent an important problem in real workshops. An approach to address this issue is to keep good performances even in presence of disturbances in workshops instead of trying to achieve exceptional performances in a deterministic status. This solution examines reaction of scheduling to disturbances after or during their generations.

Thus, the main objective of our work is to study robustness of joint production and maintenance scheduling in permutation flow shop workshops and to demonstrate that performances loss due to insertion of maintenance tasks results in a gain of joint scheduling robustness.

Given their complexity, there are few works dealing with scheduling robustness problems. The cases studied in this field are mainly those of only production scheduling. Works in Cui et al. (2004), Leus and Herroelen (2007) address the problem of finding robust scheduling for a single machine with failure uncertainty. In Cardin et al. (2013), authors carry out experimentations on a complex flexible manufacturing system in order to determine whether or not flexibility of scheduling methods can absorb uncertainties. Scheduling robustness for a flexible job shop problem with random machine failures is addressed in Cardin et al. (2013), He and Sun (2013), Weckman et al. (2012). Two-stage approaches to achieve stable and robust scheduling despite uncertain events are proposed in Alfieri et al. (2012), Huang et al. (2014), Mirabi et al. (2013), Rahmani and Heydari (2014). Authors in Ghezail et al. (2010), Rasconi et al. (2010) introduce dif-

ferent methodologies to perform a comparative evaluation of different approaches used to deal with the problem of scheduling with uncertainty. It is important to note that one of the original aspects of our contribution is that all of these works consider robustness of only production scheduling and did not study the impact of maintenance on this robustness.

The rest of this paper is organized as follows. “Literature review” section provides a focused literature review on joint production and maintenance scheduling. “Study background” section gives a background to our study describing the used data, functions, measures and their denotations. “Joint scheduling strategies” section presents the two joint production and maintenance scheduling strategies used in our study to know the sequential and the integrated strategy with the adaptation of the scheduling generating methods to each strategy. “Scheduling robustness” section describes our approach to study the scheduling robustness by presenting the considered disturbances and the procedures of applying these disturbances on both production and joint scheduling. “Experimental results” section provides experimental studies showing the merits of our approach. “Comparative Study” section presents a comparative study between our approach and other joint scheduling strategies. “Conclusion” section concludes this paper.

Literature review

In the literature, different ways of understanding and characterizing the problem of production and maintenance scheduling are proposed. Nevertheless, most works have opted for an integrated approach to solve a static problem (Ben Ali et al. 2011; Fitouhi and Nourelfath 2014; Wang and Liu 2013; Cui et al. 2014; Hadidi et al. 2011; Najid et al. 2011). In other words, maintenance and production tasks are considered simultaneously and the characteristics (availability date, operating time, etc.) of these tasks are known a priori. Some authors propose to carry out the maintenance tasks during the programmed stops of machines for other activities (quality control for example) (Ben Ali et al. 2011; Kovács et al. 2011; Fitouhi and Nourelfath 2014; Gustavsson et al. 2014; Wong et al. 2013; Roux et al. 2013; Najid et al. 2011; Tsai et al. 2001).

Other works focus on the planning level and propose a scheduling of maintenance and production operations, without necessarily worrying about operational conflicts that may arise (Buzacott and Shanthikumar 1993; Boukas et al. 1995; Rahim and Ben-Daya 2012; Nourelfath and Châtelet 2012; Wang and Liu 2013). On the other hand, authors in Moradi et al. (2011), Ben Ali et al. (2011), Kovács et al. (2011), Luo et al. (2011), Xia et al. (2015), Weinstein and H (1999), Lee and Chen (2000) deal with scheduling problems by considering all constraints to optimize a given criterion, regardless of the decision-making level.

Many studies in the literature concern single-machine problems that can form basis of complex scheduling (Wang and Liu 2013; Cui et al. 2014; Leus and Herroelen 2007; Mehta 1999; Shrouf et al. 2014; Tambe and Kulkarni 2015). Other works were interested in the scheduling problems of parallel machines that can be identical or not (Joo and Kim 2015; Lee and Chen 2000; Nourelfath and Châtelet 2012; Behnamian and Ghomi 2013; Berrichi and Yalaoui 2013; Mensendiek et al. 2015). The complexity of parallel machines scheduling prompted the majority of researchers to propose relaxed simulation of the generalized problems or some heuristics to solve them (James and Almada-Lobo 2011; Edis et al. 2013; Beraldi et al. 2008). Since the author in Johnson (1954) proposed a resolution algorithm of the flow shop model, many works considered this model (Sun et al. 2011; Ribas et al. 2010; Pan and Ruiz 2013; Neufeld et al. 2016; Yenisey and Yagmahan 2014). A part of these works has been restricted to the case of two machines given the difficulty of the general flow shop problem (Yang and Wang 2011; Espinouse et al. 2001; Zhao and Tang 2011; Mirabi et al. 2013; Huang et al. 2014). Furthermore, only few works are devoted to this model in presence of maintenance tasks, even if it has many applications in several industrial fields (Sun et al. 2011; Yenisey and Yagmahan 2014; Neufeld et al. 2016).

In the literature, many works focus on production criteria even in cases where the periods of unavailability are due to maintenance interventions (Harjunkoski et al. 2014; Moon et al. 2013; Joo and Kim 2015; Branke et al. 2016). This gap justified the orientations to give to our study concerning the consideration of maintenance when solving the production scheduling problem and the resolution strategy.

First, we consider the preventive maintenance regarding its predefined nature (Roux et al. 2013; Ruiz et al. 2007; Tsai et al. 2001; Wang and Liu 2013; Wong et al. 2013; Fitouhi and Nourelfath 2014; Gustavsson et al. 2014; Moradi et al. 2011; Naderi et al. 2011). Unlike the randomness of corrective maintenance, which acts to repair the failures, preventive maintenance results in planned interventions or periodic inspections aimed at avoiding the failures. In this case, finding the production scheduling is correlated with the resolution of the maintenance scheduling, which justify the need of a close cooperation between the departments responsible of these two functions. The literature mainly proposes two ways to choose periods of preventive maintenance: predetermined fixed periods or random periods generated during scheduling (Gertsbakh 2013; Xiong et al. 2013). For our case study, we consider predetermined periods of preventive maintenance. Nevertheless, we add tolerance intervals around each maintenance period. During these intervals, the maintenance cost is considered low, which gives more flexibility to maintenance planning.

Second, we chose a joint policy of production and maintenance scheduling. This policy allows companies to have better control of production and maintenance costs. An example of these costs are shortfalls generated by the production downtime during failures or maintenance periods (Wong et al. 2013; Hadidi et al. 2011; Rahim and Ben-Daya 2012; Tambe and Kulkarni 2015; Beraldi et al. 2008; Najid et al. 2011). Hence, this policy must be the result of a joint consultation between the two respective departments responsible of production and maintenance functions. Nevertheless, since production remains the predominant function in manufacturing systems, it will be a strong constraint for maintenance planning. Mainly, because priorities management related to the machines on which maintenance tasks must run. One of the main methods proposed to address this constraint is to take advantage of machines idle time to schedule the maximum tasks of maintenance and minimize production downtime (Naderi et al. 2011; Ruiz et al. 2007).

Third, we consider some scheduling strategies to manage operational conflicts that may arise when integrating maintenance into production scheduling. Since the functions of production and maintenance act on the same resources, namely machines, three main scheduling strategies of these two functions were identified. These strategies aim to resolve most effectively structural conflicts that may exist between the activities of production and maintenance (Campbell and Reyes-Picknell 2015; Lee and Chen 2000). The first strategy deals with an independent scheduling of these activities resulting in additional costs and delays in system operation (Tsai et al. 2001; Shahidehpour and Marwali 2012). The second strategy is to schedule jointly and sequentially the tasks of production and maintenance (Wong et al. 2013; Espinouse et al. 2001; Sloan 2004). The third strategy, called integrated, is to create a joint and simultaneous scheduling of maintenance and production tasks (Hadidi et al. 2011; Ben Ali et al. 2011; Najid et al. 2011). In our study, we opted for the two last strategies since they limit the risk of interference between production and maintenance activities and thus optimize the scheduling quality.

Study background

In this section we first describe data used in our work and their denotations, to know data of the permutation flow shop workshops and those relating to the systematic preventive maintenance as shown in Fig. 1. Then, we present the objective function used in our study.

Joint scheduling data

In this study we are interested in permutation flow shop workshops which are frequently encountered in practice. Let m be the number of machines in the workshop and n the number of

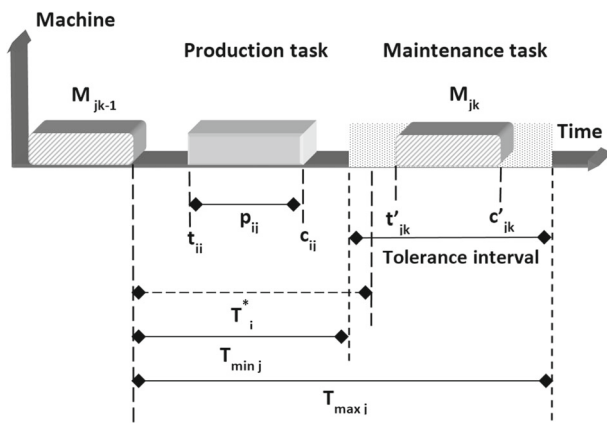


Fig. 1 Temporal characteristics of production and maintenance tasks

jobs to be processed. A job is defined as a sequence of m elementary production tasks following the order of machines. We denote by j ($1 \leq j \leq m$) a machine, i ($1 \leq i \leq n$) a job and $P(i, j)$ a production task of a job i processed on a machine j . It is also noted for each job i :

- r_i : release date (the arrival date of the job i in the workshop).
- d_i : due date (the delivery date of the job i).
- t_{ij} : starting date on a machine j .
- c_{ij} : completion date on a machine j .
- p_{ij} : processing time on a machine j .
- $Cmax_i$: $\max(c_{ij})$, the completion date of the last task of the job i .

In a permutation flow shop workshop, we consider that each machine can be subjected to one or more systematic preventive maintenance tasks which are periodic interventions. Each preventive maintenance task is characterized by some parameters predetermined by the maintenance department or by the manufacturer of the involved equipments. In practice some positive or negative deviations are tolerated about the ideal maintenance period. These deviations are supported by some tolerance intervals around each maintenance period (Benbouzid et al. 2003). In our study, the cost of a maintenance task if it is advanced or delayed within these intervals is acceptable. These intervals denoted give more flexibility to the maintenance planning if necessary and they represent a compromise between the maintenance cost and the risk of machine availability loss. In the following we denote by :

- M_j : the maintenance task associated with the machine j . A machine may be subjected to several interventions (occurrences of M_j)
- T_j^* : the optimum periodicity of the maintenance task M_j .
- $T_{min j}$: the minimum time between two successive maintenance tasks M_j on a machine j .
- $T_{max j}$: the maximum time between two successive maintenance tasks M_j on a machine j .

p'_j : the processing time of the maintenance task M_j . It is assumed to be known and constant.

The following notations are related to the k th occurrence of the maintenance task M_j :

- t'_{jk} : the start date of the k th occurrence of M_j .
 - c'_{jk} : the completion date of the k th occurrence of M_j .
- And the tolerance interval of the k th occurrence of the maintenance task M_j [T_{minjk}, T_{maxjk}] is determined as follows :

$$T_{minjk} = t'_{jk-1} + p'_j + T_{minj} \text{ and } T_{maxjk} = t'_{jk-1} + p'_j - T_{maxj}.$$

Objective function

The objective function to optimize is a compromise between goals we want to achieve for a joint scheduling of production and maintenance tasks. Several constraints imposed by customers to their suppliers often imply the minimization of the manufacturing total time or makespan C_{max} .

The respect of customers constraints imposes a proper functioning of suppliers production system which requires a planned maintenance periods. We denote by f_1 the evaluation function of this maintenance planning expressed by:

$$f_1 = \sum_{j=1}^m \sum_{k=1}^{k_j} E'_{jk} + L'_{jk} \quad (1)$$

where : k_j is the number of effective occurrences of the maintenance task M_j .

E'_{jk} : the advance of the k th occurrence of M_j .

$$E'_{jk} = \max(0, (t'_{jk-1} + p'_j + T_{minjk}) - t'_{jk})$$

L'_{jk} : the delay of the k th occurrence of M_j .

$$L'_{jk} = \max(0, t'_{jk} - (t'_{jk-1} + p'_j - T_{maxjk}))$$

To optimize the two evaluation function C_{max} and f_1 we consider the following objective function:

$$f = \alpha C_{max} + \beta f_1 \quad (2)$$

α and β are parameters that measure the respective contributions of production and maintenance in the global objective function. In case of only production scheduling, the objective function is obtained by putting the coefficient β equal to 0. The objective function is first used to generate initial joint scheduling using methods and strategies presented in "Joint scheduling strategies" Section. Then, it is used to evaluate performances deviation of these scheduling after application of disturbances according to the procedure presented in "Scheduling robustness" section.

Joint scheduling strategies

The activities of production and maintenance appear as antagonistic. Indeed, if the production has the role of operating a system to achieve certain objectives, maintenance often requires system stops to maintain it. However, the connection between these two functions is necessary in order to achieve an optimal system functioning. The objective is to respect the best equipments maintenance frequency with a minimal damage to the production plan. To do this, we will consider two options: the sequential strategy supported by two separate services and the integrated strategy supported by a single production and maintenance service. These strategies use methods of solving permutation flow shop workshops scheduling that are based on techniques of combinatorial optimization.

Traditionally, the field of combinatorial optimization has preferred the exact methods to the detriment of approximate methods. The use of accurate methods allows benefiting from strong theoretical results accompanying notions of convergence and global optimality. Thus, several classical problems (reasonably sized) can be solved exactly, in a reasonable period of time. The exact methods generally involve enumerating all solutions of the search space in order to select the best one. In addition, to improve the enumeration of solutions, exact methods have techniques to detect the earliest possible failures and specific heuristics to guide choices. However, the computation time of such methods is exponential which increases as a function of the problem size. Therefore, these methods are only usable on problems where the number of combinations is small enough to explore the solutions space in a reasonable time. As exact methods applied to production scheduling problem, we can cite linear and dynamic programming (Johnson and Montgomery 1974) and Branch and Bound algorithm (Park and Kim 2000).

Recently, there is growing interest for the ability of the approximate methods, particularly metaheuristics, to provide high quality solutions for a reasonable computation time. The loss of optimality with these methods is compensated by the decrease of calculation time and therefore an increased responsiveness. Metaheuristics also benefit from other significant advantages such as their quick adaptation to structural changes of the problem. All metaheuristics are based on a balance between intensifying research and diversifying it. On the one hand, intensification allows searching for higher quality solutions based on already found ones and on the other hand, diversification implements strategies to explore a larger solutions space and escape from local minima. We present in the following an adaptation of some joint scheduling resolution metaheuristics, that provide this balance, to the sequential and integrated strategies. Before that, we give the solution coding that we used to represent joint scheduling.

Solution coding

A joint scheduling solution describes the executing sequence of production and maintenance tasks on all machines. Therefore, each solution is encoded by two components. The first component is a sequence S which represents order of production tasks execution coding as vector of size n (number of tasks) which elements represent tasks numbers. The second component is a matrix M that represents the insertion locations of maintenance tasks. An element $M[j, k]$ of the matrix M represents the insertion location of the k th maintenance task on the j th machine according to the sequence S . For example, $M[1, 2] = 5$ means that the second maintenance task of the first machine is inserted at the 5th position in the sequence S .

Sequential strategy

We give in the following settings and algorithms of the approaches that we used in the first phase of the sequential strategy to know the production scheduling generating. Then, we present the second phase of this strategy which concerns the heuristic of maintenance tasks insertion.

Ant colony optimization

Ant colony optimization (ACO) is a metaheuristic proposed by in Dorigo (1992) for solving combinatorial problems. It has proved its efficiency in many optimization problems, especially in production scheduling (Alaykyran et al. 2007; Marimuthu et al. 2009). In the following we show our ACO implementation for generating a joint scheduling with the sequential strategy in a permutation flow shop workshop.

(1) *Pheromone information initialization* we represent pheromone information (called also intensity) in a general matrix τ of dimension $n \times n$ with n the number of tasks. A value of an element $\tau[i, r]$, such that i represents the source task and r the destination task is initialized by a real value equal to 0.1.

(2) *Heuristic information initialization* this information, called also visibility, is represented by a global matrix η of dimension $n \times n$, such that n is the number of tasks. An element $\eta[i, r]$ represents quality of a task r for an ant positioned on a task i . Elements of the matrix η are initialized by values related to the processing times of the tasks belonging to the neighbourhood of an ant a using the following formula:

$$\eta_{ir} = 1 - \frac{\sum_j p_{rj}}{\sum_{l \in N_i^a} \sum_j p_{lj}} \quad (3)$$

where:

i : current task;

p_{rj} : processing time of a task r on a machine j

l : tasks belonging to the neighbourhood N_i^a of the ant a ;

(3) *Ants positioning* each ant is positioned on a randomly chosen task which will be inserted into a Tabu list belonging to this ant. Tabu list of an ant is a vector of a maximum size equal to the number of tasks in the problem.

(4) *Solution construction (production sequence)* each ant a selects the next task to be scheduled by calculating a transition probability at a time t given by:

$$\Pi_{ir}^a = \frac{[\tau_{ir}(t)]^\alpha [\eta_{ir}(t)]^\beta}{\sum_{l \in N_i^a} [\tau_{il}(t)]^\alpha [\eta_{il}(t)]^\beta} \quad (4)$$

where: N_i^a : set of neighbours of task i not yet visited by ant a ; α and β are two parameters controlling the relative intensity τ_{ir} and the visibility η_{ir} of the track (i, r) .

Then, the selected task in the previous step is inserted in the Tabu list of the ant a . An ant moving from a task i to a task r reduces the value of the pheromone information value τ_{ir} on the track (i, r) . This update is performed in order to make less attractive the visited task to further research. The new value of τ_{ir} is calculated as following :

$$\tau_{ir}(t+1) = (1 - \rho)\tau_{ir}(t) + \rho\tau_0 \quad (5)$$

where:

ρ : the evaporation rate of pheromone with $\rho \in [0, 1]$;

τ_0 : the initial value of the pheromone on the track (i, r) . If there are still tasks to be scheduled by the ant a , then the construction process is repeated. The production sequence obtained by the ant a is the sequence contained in its Tabu list.

(5) *The best solution updating* once each ant has constructed a solution (production sequence), an update is performed to the circuit of the ant that generated the best solution. This update concerns the matrix τ (pheromone information) and it is made according to the following formula:

$$\tau_{ir}(t+1) = (1 - \alpha)\tau_{ir}(t) + \alpha\Delta\tau_{ir}(t) \quad (6)$$

where : $\Delta\tau_{ir}(t) = 1/\text{Objectivefunction}$.

(6) *Solution improvement* a kind of hybridization of the ACO algorithm and the local search heuristic (Michiels et al. 2010) is used in order to improve generated solutions and then accelerate the convergence process. In our ACO implementation, three levels of improvements are used (presented in Algorithm 1).

(7) *Stop criterion* The ant colony is restarted from step (3) while the fixed number of generations is not reached.

We give in the following the general algorithm (Algorithm 1) that illustrates the different stages of our ACO implementation.

Algorithm 1: ACO algorithm

```

begin
  Parameters initialization : Pheromone and Heuristic
  information
  while the stop criterion is not reached do
    Ants positioning
    foreach ant do
      while there are still tasks to be scheduled by the
      current ant do
        Select the next task to be scheduled
        Insert the selected task in the Tabu list
        Update the pheromone information value
      end
      Improvement level 1
    end
    Improvement level 2
    Update the best solution
  end
  Improvement level 3
end

```

Genetic algorithm

Genetic algorithm (GA) (Goldberg 1989) belongs to the class of evolutionary algorithms. These algorithms generate solutions to optimization problems using techniques inspired by natural evolution. GA has already demonstrated its performances to solve production scheduling problems (Chiou et al. 2012; Ventura and Yoon 2013). We present here steps of our GA implementation for a joint production and maintenance scheduling problem using sequential strategy.

- *Initial population generating* initial population is formed by a set of randomly generated individuals representing production sequences.
- *Adaptation function* as dealing with a problem that aims a minimization of the objective function, we chose the following adaptation value : $1/(1+\text{objectivefunction})$. The objective function is added to 1 to avoid division by zero.
- *Selection* we used the roulette wheel method that provides a selection of the best individuals by rank and tournament (Goldberg 1989).
- *Crossover* the objective is to combine two individuals called parents to make one or two new individuals called sons. In our case the k-point crossover (Goldberg 1989) is used. Its principle is to randomly generate k points of crossover which provides $k + 1$ segments in each parent.

Then to copy odd parts (respectively even parts) of the first parent in the first son (respectively the second son). The missing segments are completed by the missing tasks according to their apparition order in the second parent.

- *Mutation* it consists of swapping tasks of two randomly chosen positions of an individual according to some mutation rate.
- *Replacement* we implemented the N-Best strategy (Goldberg 1989) where the N (given as parameter) best individuals are chosen to form the new population.
- *Evaluation* adaptation values (fitness) are used to favour the best individuals. This evaluation is only conducted on new individuals selected for replacement.
- *Stopping criterion* the algorithm stops when the number of generations which is defined by the user is reached or when a stagnation of the best solution is observed.

The general functioning principle of Genetic Algorithm used with the sequential strategy is given in the following (Algorithm 2).

Algorithm 2: Genetic Algorithm

```

Data:  $P_0$  : Initial population of  $n$  production sequences
(individuals);  $n$  : The requested size of population;  $i$  :
Population index;  $S_1, S_2, S'_1, S'_2$  : Production sequences
begin
   $i \leftarrow 0$ 
  while the stop criterion is not reached do
     $i \leftarrow i + 1$ 
    Let  $P_i$  be an empty population
    while  $P_i$  does not contain  $n$  production sequences do
      Randomly select two production sequences  $S_1$  and  $S_2$ 
      from  $P_{i-1}$ 
      Apply the crossover operator on  $S_1$  and  $S_2$  to get two
      sons  $S'_1$  and  $S'_2$ 
      Add  $S'_1$  and  $S'_2$  to  $P_i$ 
    end
    Apply the mutation operator on some production
    sequences of  $P_i$ 
     $P_i \leftarrow P_i \cup P_{i-1}$  // Apply the replacement
    operator
    Apply the selection operator on  $P_i$ 
  end
  Select the best production sequence from  $P_i$ 
end
  
```

Tabu search

Tabu search (TS) is an iterative metaheuristic introduced by Glover (1986). It is very effective to solve a considerable number of combinatorial optimization problems, in particular production scheduling problem (Bozejko et al. 2013; Grabowski and Wodecki 2004). In the following, we present our adaptation of Tabu search used with the sequential strategy for solving the joint production and maintenance scheduling.

Initial solution and neighbourhood generating To generate an initial production sequence of size n , we first initializes it with a canonical solution $\{0, 1, 2, \dots, n\}$. Then, some changes are performed on this solution by randomly generating two positions and switching the tasks in these positions. The same principal is used to generate neighbourhood of a production sequences used by Tabu search algorithm (Algorithm 3).

Tabu list Tabu list allows to avoid blockages in local minima by prohibiting consideration of previous configurations. The list size is maintained throughout the method evolution. If its size is reached while a good solution is presented then the oldest solution in the list is removed to insert the new one.

Stopping criterion Search stops when a certain number of iterations is reached (defined as parameter) or when no improvement is observed for some number of iterations (stagnation). Algorithm 3 presents our adaptation of Tabu search used with the sequential strategy.

Algorithm 3: Tabu search algorithm

```

Data:  $S$  : current solution;  $S^*$  : best solution;  $S', S''$  :
intermediate solutions;  $f$  : objective function;  $NbIter$  :
Number of iterations
begin
  Randomly generate an initial solution  $S$ ; Insert  $S$  in Tabu list;
   $S^* \leftarrow S$ 
   $NbIter \leftarrow 1$ 
  while the stop criterion is not reached do
    Generate a neighbourhood  $N(S)$  of  $S$ 
     $S' \leftarrow$  the best solution of  $N(S)$ 
    if  $f(S') < f(S)$  then
       $S \leftarrow S'$ 
       $S^* \leftarrow S$ 
      Insert  $S'$  in Tabu list
    else
      Randomly take  $S''$  out of Tabu list
       $S \leftarrow S''$ 
      Increment stagnation parameter
    end
     $NbIter \leftarrow NbIter + 1$ 
  end
  
```

Maintenance tasks insertion

The principle of maintenance tasks insertion in a production sequences obtained by one of the previous algorithms is to process all maintenance tasks of the first machine, then we move to the second and so on until the last machine according to the depth-first search algorithm (Kleinberg et al. 2006). This strategy does not bear risks of missing tasks and has the advantage that an inserted maintenance task will no longer be shifted.

Insertion of the k th occurrence of a maintenance task on a machine j is done at locations that minimize a partial objective function (a partial evaluation of the obtained joint scheduling) according to the following cases:

- Within its tolerance interval $[T_{minj}, T_{maxj}]$. In this case, there is no delay of a maintenance task.
- Before its T_{minjk} . To avoid having a large number of maintenance tasks too close, this insertion is possible only if T_{minjk} does not exceed a margin $\tau \times T_{minj}$ ($0 < \tau < 1$): $T_{minjk} - t'_{jk} \leq \tau \times T_{minj}$.
- After its T_{maxjk} (delayed task). To ensure that no maintenance task will be missed, this insertion is possible only if its delay does not exceed a margin $\tau \times T_{maxj}$ ($0 < \tau < 1$): $t'_{jk} - T_{maxjk} \leq \tau \times T_{maxj}$.

Integrated strategy

The integrated strategy consists on ordering production and maintenance tasks at the same time, while optimizing the overall objective function. In the following, we present the integrated versions of the same approaches used with the sequential strategy.

Integrated ant colony optimization

Steps for generating an *ACO* joint scheduling using the integrated strategy are the same as those of the sequential strategy. The main modifications concern the solution construction and the best solution updating. These modifications are given in the following.

Construction of a joint solution (production and maintenance sequence) In this step, each ant builds a production sequence using the same procedure described in the sequential strategy. Then maintenance tasks are inserted using the depth-first search algorithm after each production sequence generating by each ant. The insertion of maintenance tasks at this level allows the evaluation to be done on the joint sequence and not only on the production sequence. The best solution is the joint scheduling that optimizes (minimizes) the overall objective function.

The best solution updating Once each ant in the colony built a solution (joint scheduling), an update is performed to the circuit of the ant which obtained the best solution at each generation. The considered best solution here is the one that minimizes the overall objective function given in formula (2).

Integrated genetic algorithm

The main difference between GA using the integrated strategy and GA using the sequential strategy is that, with the

integrated strategy, the crossover and mutation operators consider at the same time production and maintenance tasks. In the following, we present the principle of these operators.

Crossover The final crossover operators are obtained by combining crossovers operators on production and those on maintenance. Crossover operators on production tasks are those defined for the sequential strategy and presented in “Genetic algorithm” section. The new crossover operator on maintenance is inspired by the k -points crossover operator principle. It consists on generating k ($k \in [0 \dots m]$) random points of crossover on each parent (this provides $k + 1$ segments in each parent). Then, copying maintenance tasks located in the even parts (respectively odd parts) of the first parent into the even parts (respectively odd parts) of the first son (respectively the second son). In the same way, copying maintenance tasks located in the odd parts (respectively even parts) of the second parent into the odd parts (respectively even parts) of the first son (respectively the second son).

Mutation The mutation operator on production follows the same principle as that of the sequential strategy. At the same time, locations of maintenance tasks will be changed relatively to the new locations of the production tasks. In addition, randomly shifts to the left, or to the right, are applied on some maintenance tasks.

Integrated tabu search

In the following, we present the main changes of the integrated Tabu search algorithm compared to that used with the sequential strategy.

Initial solution generating To obtain the initial solution, Tabu search began by randomly generating a production sequence using the same principal presented in “Tabu search” section. Then, it inserts maintenance tasks in the obtained production sequence using the heuristic presented in “Maintenance tasks insertion” section.

Neighbourhood generating We present in the following two principles of movements used to generate the neighbourhood solutions :

1. Neighbourhood generating by maintenance tasks shifting: a maintenance task can have several possible locations in its tolerance interval. Therefore, it is useful to generate the neighbourhood of a solution using some possible locations that one or more maintenance tasks can have on one or more machines.
2. Neighbourhood generating by production tasks swapping: the purpose of this operation is to create new individuals by changing the execution order of production tasks groups using the same principle presented in

“Tabu search” section, but keeping the right maintenance tasks location in relation with production tasks.

Hybrid methods

This trend of solutions aimed hybridization of optimization methods by making two or more of these methods working together to achieve the best solutions (Costa et al. 2015; Laha and Chakraborty 2009; Meeran and Morshed 2012). In our study, a hybridization between methods mentioned above is tried. Hybridizations are performed on the two joint scheduling strategies, the sequential and the integrated one. In the following, we propose two hybrid methods used in our study.

GA hybridization with tabu search (GAH) GA algorithm generates an initial solution for Tabu search. Then, Tabu search try to improve this solution.

ACO hybridization with tabu search (ACOH) The same precedent principle is applied, Tabu search starts with an initial solution generated by the ACO algorithm, then try to improved it.

Parameter tuning

Many parameters have to be tuned for the metaheuristics used in our study. In literature, there are two different strategies for parameter tuning : the off-line parameter initialization (or meta-optimization) and the online parameter tuning strategy (Talbi 2009). In off-line parameter initialization, the values of different parameters are fixed before the execution of the metaheuristic, while in the online approach, the parameters are updated dynamically during the execution of the metaheuristic. In our experimentations, we used the off-line tuning method “irace” (López-Ibáñez et al. 2011), a publicly available implementation of Iterated F-Race proposed by Balaprakash et al. (2007) and further developed by Birattari et al. (2010). The main purpose of irace is to automatically configure optimization algorithms by finding the most appropriate settings given a set of tuning instances of an optimization problem. Irace starts by sampling a number of parameter settings of a given algorithm uniformly at random. Then, at each iteration, it selects a set of best settings using a racing procedure (Maron and Moore 1997) and Friedman’s non-parametric two-way analysis of variance by ranks (Daniel et al. 1990). This racing procedure iteratively runs the algorithm settings on a sequence of problem instances, and rejects settings as soon as there is enough statistical evidence that they perform worse than the best one. Then, the best settings are used to bias a local sampling model. The next iteration begins by selecting new settings from this model, then it applies the racing procedure to these settings with the previous best settings. This procedure continues to run

until a given computational budget is exhausted. This computational budget may be an overall computation time or a number of experiments, where an experiment is the application of a configuration to an instance. In our study, we tuned each optimization algorithm (ACO, GA and TS) using a budget of 5000 experiments.

Scheduling robustness

In literature of scheduling in dynamic and stochastic environments, the robustness of a solution usually refers to a scheduling prepared for uncertainties of future unknown events (Cowling and Johansson 2002; Jensen 2001; Yellig and Mackulak 1997). The uncertainties may be from changes in the environment or from inaccurate execution of the solution itself. This is why the authors in Billaut et al. (2013), Pinedo (2012) noted that the robustness is a difficult concept to measure or even to define. Some authors prefer to use the concept of robust methods to describe a method that provides scheduling which preserves its performances in the presence of disturbances (Daniels and Kouvelis 1995; Shafaei and Brunn 1999).

The notion of disturbances here, is defined generally as any unanticipated operational event introduced during the implementation of the scheduling and which can affect its performances (Cowling and Johansson 2002; Mehta 1999; Shafaei and Brunn 1999). The disturbances considered in our study concern extensions of tasks processing time according to some classes and levels. These extensions can be justified by the need of having a better product quality or by consequences of eventual machines failures.

In literature, there are several approaches proposed to handle disturbances when solving optimization problems, a part of these approaches are cited in Bertsimas et al. (2011). However, a practical drawback of these approaches is that they are not computationally tractable for some optimization problems where the number of possible combinations is important (case of flow shop scheduling when considering a significant number of machines and tasks). In addition, these approaches generally have some cost in terms of optimality associated with robustness (Bertsimas and Sim 2004).

In the following, we give some necessary definitions for designing a model to study the robustness of joint production and maintenance scheduling in the presence of disturbances. In this model, we generate and apply disturbances on both production and joint scheduling. Our approach doesn’t handle disturbances when generating scheduling but it evaluates the robustness of the generated scheduling using algorithms presented in “Joint scheduling strategies” section. Our approach can also be adapted to study the robustness of joint scheduling generated using other metaheuristics or exact methods.

The disturbances classes represent the rate of tasks assigned to the machines subjected to disruptions (which will fail for example). Three classes intervals noted c_i ($c_i \in \{ [0, 40\%], [40, 70\%], [70, 100\%] \}$) are considered. The disturbances rate U_c is randomly chosen following a uniform distribution in the interval c_i . $U_c \sim U[a_i, b_i]$ with a_i and b_i are the interval bounds.

Disturbances levels are used to calculate the maximum extending processing time rate of tasks subjected to disturbances. Ten intervals noted l_i are considered. The extending rate is randomly chosen following a uniform distribution in the interval l_i ($l_i \in \{ [0, 10\%], [10, 20\%], \dots, [90, 100\%] \}$). $U_l \sim U[\alpha_i, \beta_i]$ with α_i and β_i are the interval bounds.

Assuming that there are n production tasks then the total number of tasks to disturb NT is given by the following formula:

$$NT = [n * U_c] \quad (7)$$

where $[i]$ is the integer part of i .

Note that a total number of tasks to disrupt NT equal to 0 can be obtained using the latter formula. To avoid this case, NT is considered to be at least equal to 1.

On the other hand, a disturbance level is used to specify the maximum extension rate of production tasks processing times. New processing times of tasks subjected to disturbances is calculated using the following formula:

$$p''_{ij} = p_{ij} * (1 + \xi_{ij}) \quad (8)$$

where :

- p''_{ij} : the new processing time of the disturbed task $P(i, j)$.
 - p_{ij} : the initial processing time before disturbing the task $P(i, j)$.
 - ξ_{ij} : the disturbance coefficient of the task $P(i, j)$.
- For production scheduling, before inserting maintenance tasks, ξ_{ij} is equal to U_l .

For joint production and maintenance scheduling, the calculation of ξ_{ij} is more complicated. In fact, the insertion of maintenance tasks in the production scheduling may reduce the machines failures. Since the machines reliability does not last a long time, the risk of failure increase, the further one gets away from the last maintenance intervention. Thus, the evolution of the failure (disturbance) intensity in a period of time bounded by two successive maintenance operations on the same machine can be divided into two intervals. A first interval where the failure intensity is nil, which means that no disruption application is allowed since the machine is just subjected to a maintenance operation. This first interval is denoted by R_j for a given machine j . A second interval where the risk of a machine failure increases progressively

as her reliability decreases. This interval is called the disturbance interval and denoted by D_{jk} for a given maintenance occurrence k on a machine j .

Within the second interval, calculation of disturbances coefficient ξ_{ij} depends on the level of disturbances (the maximum disturbance intensity to be considered) and the the distance of the production task from the last maintenance task performed on the same machine. For this reason it was assumed to divide the disturbance interval D_j in ten subdivisions. These subdivisions are used to realize the effect of maintenance tasks on production tasks.

If we consider these dependencies, the value of ξ_{ij} in case of a joint scheduling is given by the following formula:

$$\xi_{ij} = \chi_{ij} * U_l \quad (9)$$

where χ_{ij} depends on the subdivision in which the task $P(i, j)$ is located. In the following we present the steps of finding the formula used to calculate its value: First, the duration ρ between the starting date t_{ij} of the task $P(i, j)$ and the end date of the reliability interval R_j , is calculated by the following formula:

$$\rho = t_{ij} - (c'_{jk} + R_j)$$

Now, if we denote by I_{jk} the time between two successive maintenance tasks on the same machine j (between the k th and the $(k + 1)$ th occurrences) then the duration of the disturbance interval D_{jk} is given by the following formula : $D_{jk} = I_{jk} - R_j$.

By performing a three-rule, the number of subdivisions x in ρ may be calculated, in order to find the subdivision in which the task $P(i, j)$ is located as follows : $x = \rho * 10 / D_{jk}$.

Finally, by substituting x in the formula $\chi_{ij} = x / 10$, we will have:

$$\chi_{ij} = (t_{ij} - (c'_{jk} + R_j)) / (I_{jk} - R_j)$$

Thus, the general formula of the disturbance coefficient is given by :

$$\xi_{ij} = \frac{t_{ij} - (c'_{jk} + R_j)}{D_{jk} - R_j} * U_l \quad (10)$$

The application of disturbances to the scheduling causes some problems due to the new coordinates of production tasks, which do not respect the nature of the workshop and the precedence constraints between tasks. In other words, overlaps, which are defined as an interlacing or a superposition of two tasks, have a great chance to appear. Two types of overlaps can be encountered : horizontal overlaps (between two tasks in succession on the same machine) and vertical overlap (between tasks of the same job).

To address this problem, we opted for the right shift approach (Vieira et al. 2003), which is a reactive approach

to solve the problem of overlaps while maintaining the tasks order. This order should be maintained to make a valid comparison between performances of the initial scheduling and performances of the disturbed ones.

If the task concerned by an overlap is a maintenance task, two situations arise. If its shifting does not cause exceeding of its T_{max} , then this shifting will be performed. Else it will be switched with the production task that caused the overlap. The switched production task will not be disturbed since it will be located just after the maintenance task, i.e. in the reliability interval. The starting date of the maintenance task will be chosen so that it does not come before its T_{min} .

Since the calculation of the new processing times after disturbances uses a deterministic function (formula (10)), the complexity of our approach is mainly related to the right shift algorithm when overlaps occur. Right shift rescheduling means simply to delay the entire scheduling by the duration of the overlap. Since this is done by adding a fixed increment of time to each tasks of the scheduling, the right shift algorithm is clearly polynomial, since the computation only involves the addition of a time increment to each of the remaining operations. At most $m \cdot n$ operations are concerned by this shift.

The general principle of disruption is given in the Algorithm 4.

Algorithm 4: Disruption scheduling algorithm

```

Data:  $n$  : number of tasks;  $m$  : number of machines;  $U_c$  : rate of
tasks that will be disturbed;  $U_l$  : extension rate of
processing times;  $\alpha_i, \beta_i, a_i, b_i$  : interval bounds;
begin
  foreach disturbances class interval  $c_i = [a_i, b_i]$  do
    foreach disturbances level interval  $l_i = [\alpha_i, \beta_i]$  do
      Generate  $U_c \sim U[a_i, b_i]$ 
      for  $nt \leftarrow 1$  to  $[n * U_c]$  do
        Generate  $U_l \sim U[\alpha_i, \beta_i]$ 
        Generate  $i \sim U[0, n]$ 
        Generate  $j \sim U[0, m]$ 
        Locate  $M_{jk}$  // the last occurrence of
the maintenance task (the  $k$ th
occurrence on the  $j$ th machine)
        if  $t_{ij} > (c'_{jk} + R_j)$  then
           $\xi_{ij} = \frac{t_{ij} - (c'_{jk} + R_j)}{l_{jk} - R_j} * U_l$ 
           $p'_{ij} = p_{ij} * (1 + \xi_{ij})$  // Calculate the
new processing time of the task
           $P(i, j)$ 
          Update scheduling with consideration of the
new processing time and overlaps
        end
      end
      Calculate the performances measures of the disturbed
scheduling
    end
  end
end

```

Experimental results

To implement the model presented in the previous sections, a sample of 20 initial scheduling is generated by each method and strategy presented in “Joint scheduling strategies” section. Each initial scheduling is disturbed according to 10 levels and 3 classes. For each class and level, 100 simulations are performed and the averages results of these 100 simulations are considered. The differences between the objective functions values of the disturbed scheduling and the initial scheduling (before application of disturbances) are considered to evaluate the performances deviations of these scheduling.

Benchmarks of Taillard (1993) and Reeves (1995) are used as production data. Maintenance data are randomly generated using a specific service developed to perform this study.

The experimental studies presented in this section try to answer the two essential questions of our study, to know : are there robust scheduling methods? Does maintenance contribute to the robustness of joint scheduling? In the following we present the experimental results according to these two questions.

Are there robust scheduling methods?

Figure 2 shows the performances deviations averages of the production scheduling calculated from the combined levels and classes of disturbances according to the used method and the workshop size. The main remark, is that methods robustness is inherent with small benchmarks for the most cases of the disturbance protocol. Moreover, the results of hybrid methods, ACO and GA are generally better than those of Tabu search.

Figure 3 shows the performances deviations averages of the joint scheduling obtained from the combined levels and classes of disturbances using the sequential strategy accord-

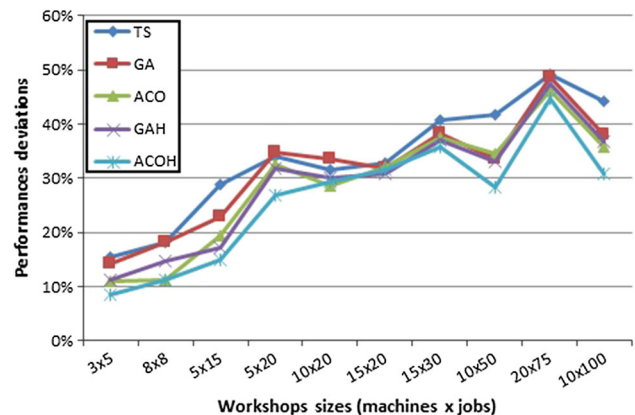


Fig. 2 Performances deviations averages of the production scheduling methods

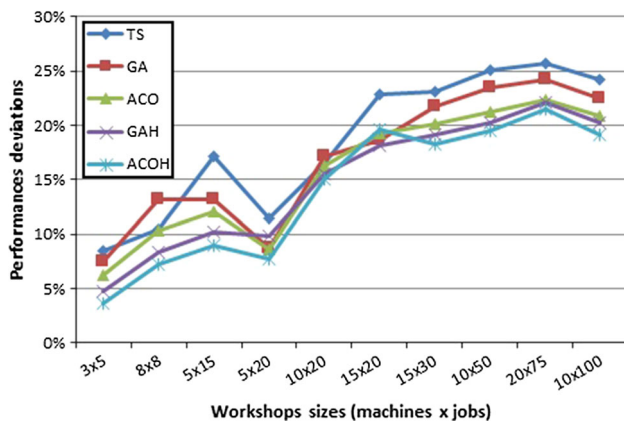


Fig. 3 Performances deviations averages of the joint scheduling methods using the sequential strategy

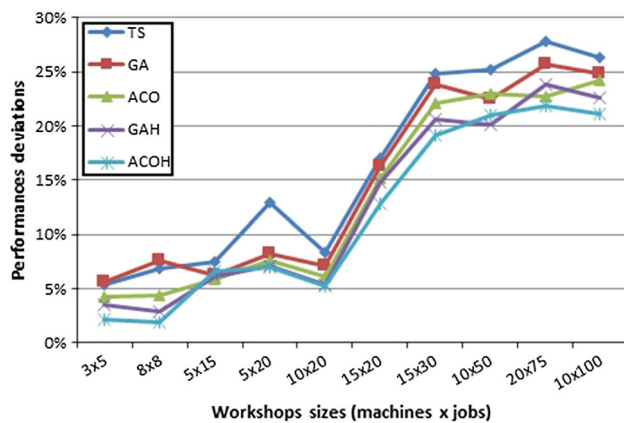


Fig. 4 Performances deviations averages of the joint scheduling methods using the integrated strategy

ing to the initial scheduling method and the workshop size. The general trend of the curves is towards a full influence of machines and tasks number. We note again that generally, results of small benchmarks are better than those of big ones. Furthermore, in the first position we have the results of the hybrid methods which are better than those of the evolutionary methods (ACO and GA). In the last position we have results of the iterative method TS.

Figure 4 shows the performances deviations averages of the joint scheduling using the integrated strategy calculated from the combined levels and classes of disturbances according to the initial scheduling methods and the workshop size. Again, the conclusion remains the same, since the robustness of scheduling methods is related to the disturbance protocol (level, class and size of the studied problems). Thus, the general trend remains towards results deterioration which increases by raising the workshop size. Although, the best results are those given by the hybrid methods followed by the evolutionary methods (ACO and GA) which give better results than the iterative method TS.

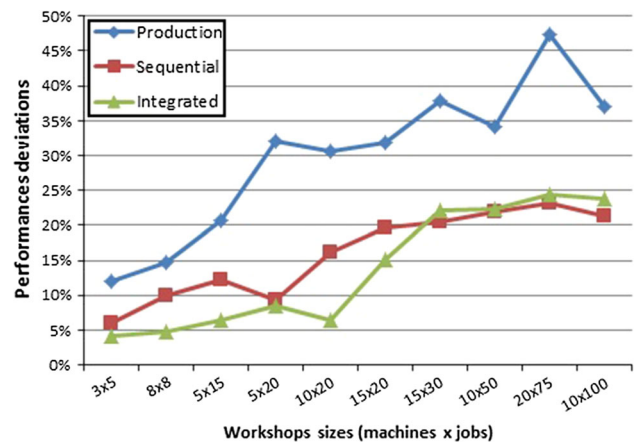


Fig. 5 Performances deviations averages of joint and production scheduling

Does maintenance contribute to the robustness of joint scheduling?

The objective is to demonstrate that, when disturbances occur, performances loss of joint production and maintenance scheduling is less than performances loss of only production scheduling. Figure 5 shows the performances deviations averages of the joint scheduling strategies compared to those of the production scheduling obtained from the combined levels and classes of disturbances of all scheduling methods presented in “Joint scheduling strategies” section.

The first observation concerns the impact of maintenance on robustness of scheduling. If we take any workshop (small or big), we note a clear performance stability of joint production and maintenance scheduling compared to only production scheduling. On average, 75% of disrupted production scheduling (different levels and classes) have a deviation over than 30%. Another remark is that for small workshops, scheduling using the integrated strategy generally gives the best results. Also, it is important to note that the results of small workshops (sequential and integrated strategy) are better than those of the big ones because of the high density of these latter on tasks and machines.

As conclusion of this part, we can say that the insertion of maintenance tasks in production scheduling is certainly done at the expense of performance of these later (while remaining in very acceptable limits), but it provides a gain in terms of robustness. Indeed the contribution of maintenance is located at the reliability intervals of the maintained machines which allows the absorption of some disturbances. Furthermore, the general trend is the same for both strategies, however, the generated scheduling by the integrated strategy are more robust than those generated by the sequential strategy if we consider the small workshops.

Comparative study

This section gives a comparative study between the robustness of our joint scheduling strategies using hybridization of ACO and Tabu search approach (which gave the best results in “Experimental results” section) and other strategies used to generate production and maintenance scheduling. In the following, we denote by ACOHSEQ and ACOHINT, our sequential and integrated strategy using hybridization of ACO and Tabu search approach. This study uses the same experimental conditions presented in “Experimental results” section.

The first strategy considered in our study, called SALPT, was presented in [Allaoui and Artiba \(2004\)](#). To develop this strategy, the authors used the simulated annealing heuristic ([Černý 1985](#)), a dispatching rule that sequences the jobs in a decreasing order of the total processing times and a flexible simulation model. The second strategy of our study was presented in [Ruiz et al. \(2007\)](#). This strategy, called PACO, is an adaptation of the ant colony optimization algorithm to implicitly consider preventive maintenance operations in a permutation flow shop problem. The proposed preventive maintenance policy aims to maintain a minimum level of reliability after the production period. The third strategy was proposed in [Wong et al. \(2013\)](#) and considers multiple resources and preventive maintenance tasks in traditional production scheduling. This strategy, called joint scheduling (JS), adopts the multi-resource maintenance modelling approach where the starting times of the preventive tasks were treated as decision variables. The new problem is formulated as an integer-programming model and is solved by the genetic algorithm approach.

Figure 6 shows the performances deviations averages of the obtained joint scheduling calculated from the combined levels and classes of disturbances according to the used strategy and the workshop size. The main remark is that

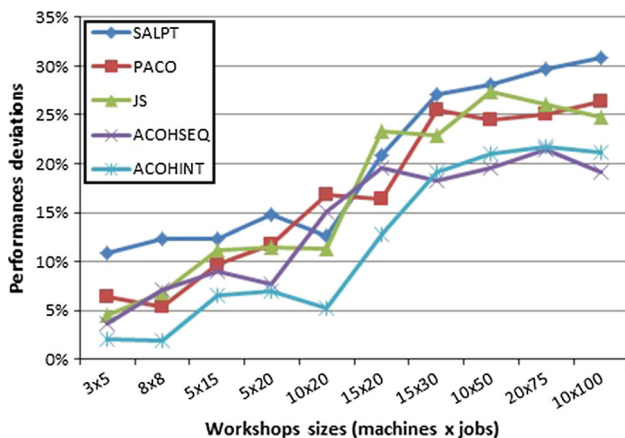


Fig. 6 Comparative study between joint scheduling strategies

the robustness of joint scheduling strategies is related to the considered number of machines and tasks. Thus, the general trend of the curves is towards results deterioration which increases by raising the workshop density on tasks and machine. Although, the hybrid method using our integrated strategy (ACOHINT) gives better results than the other strategies.

Another observation concerns the impact of maintenance on robustness of scheduling. If we consider the results presented in Figs. 2 and 6, whatever the considered strategy, performance loss of disturbed joint scheduling is less than performance loss of only production scheduling.

Conclusion

This paper proposed a study of joint production and maintenance scheduling robustness in a permutation flow shop workshop using two strategies : sequential and integrated. As methods of scheduling resolution, we proposed an adaptation of Tabu Search, Genetic Algorithm, Ants Colony Optimization and some hybridizations of these methods.

Since in the real industrial workshops, the generated joint scheduling of production and maintenance tasks will be inevitably disrupted, we proposed a new approach to study the robustness of these scheduling. Our approach considers disturbances on the processing time of production tasks according to some levels and classes. Several experimental studies were conducted in this paper. Results of these experimentations show first the merits of maintenance on production scheduling robustness. Then, the comparative study of all studied methods showed that the integrated strategy is more robust than the sequential one, and that the hybrid and the evolutionary approaches generally give the best results.

Future work is to investigate the robustness of other scheduling methods with other workshops models such as job shop and open shop ([Blazewicz et al. 1996](#); [Malakooti 2013](#)). For the scheduling robustness study, we can define another protocol and type of disturbances.

References

Alaykyran, K., Engin, O., & Doyen, A. (2007). Using ant colony optimization to solve hybrid flow shop scheduling problems. *The International Journal of Advanced Manufacturing Technology*, 35(5–6), 541–550.

Alferi, A., Tolio, T., & Urgo, M. (2012). Robust and stable flow shop scheduling with unexpected arrivals of new jobs and uncertain processing times. *The International Journal of Advanced Manufacturing Technology*, 62, 279–290.

Allaoui, H., & Artiba, A. (2004). Integrating simulation and optimization to schedule a hybrid flow shop with maintenance constraints. *Computers & Industrial Engineering*, 47(4), 431–450.

- Balaprakash, P., Birattari, M., & Stützle, T. (2007). Improvement strategies for the f-race algorithm: Sampling design and iterative refinement. In *International workshop on hybrid metaheuristics*, (pp. 108–122). Springer.
- Behnamian, J., & Ghomi, S. F. (2013). The heterogeneous multi-factory production network scheduling with adaptive communication policy and parallel machine. *Information Sciences*, 219, 181–196.
- Ben Ali, M., Sassi, M., Gossa, M., & Harrath, Y. (2011). Simultaneous scheduling of production and maintenance tasks in the job shop. *International Journal of Production Research*, 49(13), 3891–3918.
- Benbouzid, F., Varnier, C., & Zerhouni, N. (2003). Resolution of joint maintenance/production scheduling by sequential and integrated strategies. In: *Artificial neural nets problem solving methods*, (pp. 782–789). Springer.
- Beraldi, P., Ghiani, G., Grieco, A., & Guerriero, E. (2008). Rolling-horizon and fix-and-relax heuristics for the parallel machine lot-sizing and scheduling problem with sequence-dependent set-up costs. *Computers & Operations Research*, 35(11), 3644–3656.
- Berrichi, A., & Yalaoui, F. (2013). Efficient bi-objective ant colony approach to minimize total tardiness and system unavailability for a parallel machine scheduling problem. *The International Journal of Advanced Manufacturing Technology*, 68(9–12), 2295–2310.
- Bertsimas, D., & Sim, M. (2004). The price of robustness. *Operations Research*, 52(1), 35–53.
- Bertsimas, D., Brown, D. B., & Caramanis, C. (2011). Theory and applications of robust optimization. *SIAM Review*, 53(3), 464–501.
- Billaut, J. C., Moukrim, A., & Sanlaville, E. (2013). *Flexibility and robustness in scheduling*. Hoboken: Wiley.
- Birattari, M., Yuan, Z., Balaprakash, P., & Stützle, T. (2010). F-race and iterated f-race: An overview. In: *Experimental methods for the analysis of optimization algorithms*, (pp. 311–336). Springer.
- Blazewicz, J., Ecker, K., Pesch, E., Schmidt, G., & Weglarz, J. (1996). *Scheduling computer and manufacturing processes*. Berlin: Springer.
- Boukas, E. K., Zhang, Q., & Yin, G. (1995). Robust production and maintenance planning in stochastic manufacturing systems. *IEEE Transactions on Automatic Control*, 40(6), 1098–1102.
- Bozejko, W., Pempera, J., & Smutnicki, C. (2013). Parallel tabu search algorithm for the hybrid flow shop problem. *Computers and Industrial Engineering*, 65(3), 466–474.
- Branke, J., Nguyen, S., Pickardt, C. W., & Zhang, M. (2016). Automated design of production scheduling heuristics: A review. *IEEE Transactions on Evolutionary Computation*, 20(1), 110–124.
- Buzacott, J. A., & Shanthikumar, J. G. (1993). *Stochastic models of manufacturing systems* (Vol. 4). Englewood Cliffs: Prentice Hall.
- Campbell, J. D., & Reyes-Picknell, J. V. (2015). *Uptime: Strategies for excellence in maintenance management*. Boca Raton: CRC Press.
- Cardin, O., Mebarki, N., & Pinot, G. (2013). A study of the robustness of the group scheduling method using an emulation of a complex fms. *International Journal of Production Economics*, 146(1), 199–207.
- Černý, V. (1985). Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1), 41–51.
- Chiou, C., Chen, W., Liu, C., & Wu, M. (2012). A genetic algorithm for scheduling dual flow shops. *Expert Systems with Applications*, 39(1), 1306–1314.
- Costa, A., Cappadonna, F. A., & Fichera, S. (2015). A hybrid genetic algorithm for minimizing makespan in a flow-shop sequence-dependent group scheduling problem. *Journal of Intelligent Manufacturing*. doi:10.1007/s10845-015-1049-1
- Cowling, P., & Johansson, M. (2002). Using real time information for effective dynamic scheduling. *European Journal of Operational Research*, 139(2), 230–244.
- Cui, W., Lu, Z., & Pan, E. (2004). The lorentz transformation and absolute time. *Computers and Operations Research*, 47, 81–91.
- Cui, W. W., Lu, Z., & Pan, E. (2014). Integrated production scheduling and maintenance policy for robustness in a single machine. *Computers & Operations Research*, 47, 81–91.
- Daniel, W. W., et al. (1990). *Applied nonparametric statistics*. Boston: PWS-Kent.
- Daniels, R. L., & Kouvelis, P. (1995). Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management Science*, 41(2), 363–376.
- Dorigo, M. (1992). Optimization, learning and natural algorithms. PhD thesis, Politecnico di Milano, Italie.
- Edis, E. B., Oguz, C., & Ozkarahan, I. (2013). Parallel machine scheduling with additional resources: Notation, classification, models and solution methods. *European Journal of Operational Research*, 230(3), 449–463.
- Espinouse, M., Formanowicz, P., & Penz, B. (2001). Complexity results and approximation algorithms for the two machines no-wait flow-shop with limited machine availability. *Journal of the Operational Research Society*, 52, 116–121.
- Fitouhi, M. C., & Noureldath, M. (2014). Integrating noncyclical preventive maintenance scheduling and production planning for multi-state systems. *Reliability Engineering & System Safety*, 121, 175–186.
- Gen, M., & Lin, L. (2014). Multiobjective evolutionary algorithm for manufacturing scheduling problems: State-of-the-art survey. *Journal of Intelligent Manufacturing*, 25(5), 849–866.
- Gertsbakh, I. (2013). *Reliability theory: With applications to preventive maintenance*. Berlin: Springer.
- Ghezail, F., Pierreval, H., & Hajri-Gabouj, S. (2010). Analysis of robustness in proactive scheduling: A graphical approach. *Computers and Industrial Engineering*, 58(2), 193–198.
- Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 13(5), 533–549.
- Goldberg, D. (1989). *Genetic algorithms in search, optimization, and machine learning*. Salt Lake City: Addison-Wesley Professional.
- Grabowski, J., & Wodecki, M. (2004). A very fast tabu search algorithm for the permutation flow shop problem with makespan criterion. *Computers and Operations Research*, 31(11), 1891–1909.
- Gustavsson, E., Patriksson, M., Strömberg, A. B., Wojciechowski, A., & Önnheim, M. (2014). Preventive maintenance scheduling of multi-component systems with interval costs. *Computers & Industrial Engineering*, 76, 390–400.
- Hadidi, L. A., Al-Turki, U. M., & Rahim, A. (2011). Integrated models in production planning and scheduling, maintenance and quality: A review. *International Journal of Industrial and Systems Engineering*, 10(1), 21–50.
- Harjunkoski, I., Maravelias, C. T., Bongers, P., Castro, P. M., Engell, S., Grossmann, I. E., et al. (2014). Scope for industrial applications of production scheduling models and solution methods. *Computers & Chemical Engineering*, 62, 161–193.
- He, W., & Sun, D. (2013). Scheduling flexible job shop problem subject to machine breakdown with route changing and right-shift strategies. *The International Journal of Advanced Manufacturing Technology*, 66(1–4), 501–514.
- Huang, R. H., Yu, S. C., & Kuo, C. W. (2014). Reentrant two-stage multiprocessor flow shop scheduling with due windows. *The International Journal of Advanced Manufacturing Technology*, 71(5–8), 1263–1276.
- James, R. J., & Almada-Lobo, B. (2011). Single and parallel machine capacitated lot-sizing and scheduling: New iterative mip-based neighborhood search heuristics. *Computers & Operations Research*, 38(12), 1816–1825.
- Jensen, M. T. (2001). Improving robustness and flexibility of tardiness and total flow-time job shops using robustness measures. *Applied Soft Computing*, 1(1), 35–52.

- Johnson, L. A., & Montgomery, D. C. (1974). *Operations research in production planning, scheduling, and inventory control* (Vol. 6). New York: Wiley.
- Johnson, S. M. (1954). Optimal two-and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly*, 1(1), 61–68.
- Joo, C. M., & Kim, B. S. (2015). Hybrid genetic algorithms with dispatching rules for unrelated parallel machine scheduling with setup time and production availability. *Computers & Industrial Engineering*, 85, 102–109.
- Kleinberg, J., & Tardos, E. (2006). *Algorithm design*. Salt Lake City: Addison Wesley.
- Kovács, A., Erdős, G., Viharos, Z. J., & Monostori, L. (2011). A system for the detailed scheduling of wind farm maintenance. *CIRP Annals-Manufacturing Technology*, 60(1), 497–501.
- Laha, D., & Chakraborty, U. K. (2009). An efficient hybrid heuristic for makespan minimization in permutation flow shop scheduling. *The International Journal of Advanced Manufacturing Technology*, 44(5–6), 559–569.
- Lee, C. Y., & Chen, Z. L. (2000). Scheduling jobs and maintenance activities on parallel machines. *Naval Research Logistics*, 47, 145–165.
- Leus, R., & Herroelen, W. (2007). Scheduling for stability in single-machine production systems. *Journal of Scheduling*, 10(3), 223–235.
- López-Ibáñez, M., Dubois-Lacoste, J., Stützle, T., & Birattari, M. (2011). The irace package, iterated race for automatic algorithm configuration. Technical Report, TR/IRIDIA/2011-004, IRIDIA, Universit Libre de Bruxelles, Belgium.
- Luo, H., Huang, G. Q., Feng Zhang, Y., & Yun Dai, Q. (2011). Hybrid flowshop scheduling with batch-discrete processors and machine maintenance in time windows. *International Journal of Production Research*, 49(6), 1575–1603.
- Malakooti, B. (2013). *Operations and production systems with multiple objectives*. Hoboken: Wiley.
- Marimuthu, S., Ponnambalam, S., & Jawahar, N. (2009). Threshold accepting and ant-colony optimization algorithms for scheduling m-machine flow shops with lot streaming. *Journal of Materials Processing Technology*, 209(2), 1026–1041.
- Maron, O., & Moore, A. W. (1997). The racing algorithm: Model selection for lazy learners. In: *Lazy learning*, (pp. 193–225). Springer.
- Meeran, S., & Morshed, M. (2012). A hybrid genetic tabu search algorithm for solving job shop scheduling problems: A case study. *Journal of Intelligent Manufacturing*, 23(4), 1063–1078.
- Mehta, S. V. (1999). Predictable scheduling of a single machine subject to breakdowns. *International Journal of Computer Integrated Manufacturing*, 12(1), 15–38.
- Mensendiek, A., Gupta, J. N., & Herrmann, J. (2015). Scheduling identical parallel machines with fixed delivery dates to minimize total tardiness. *European Journal of Operational Research*, 243(2), 514–522.
- Michiels, W., Aarts, E., & Korst, J. (2010). *Theoretical Aspects of Local Search*. Berlin: Springer.
- Mirabi, M., Fatemi Ghomi, S. M. T., & Jolai, F. (2013). A two-stage hybrid flowshop scheduling problem in machine breakdown condition. *Journal of Intelligent Manufacturing*, 24(1), 193–199.
- Moon, J. Y., Shin, K., & Park, J. (2013). Optimization of production scheduling with time-dependent and machine-dependent electricity cost for industrial energy efficiency. *The International Journal of Advanced Manufacturing Technology*, 68(1–4), 523–535.
- Moradi, E., Ghomi, S. F., & Zandieh, M. (2011). Bi-objective optimization research on integrated fixed time interval preventive maintenance and production for scheduling flexible job-shop problem. *Expert Systems with Applications*, 38(6), 7169–7178.
- Naderi, B., Zandieh, M., & Aminnayeri, M. (2011). Incorporating periodic preventive maintenance into flexible flowshop scheduling problems. *Applied Soft Computing*, 11(2), 2094–2101.
- Najid, N. M., Alaoui-Selsouli, M., & Mohafid, A. (2011). An integrated production and maintenance planning model with time windows and shortage cost. *International Journal of Production Research*, 49(8), 2265–2283.
- Neufeld, J. S., Gupta, J. N., & Buscher, U. (2016). A comprehensive review of flowshop group scheduling literature. *Computers & Operations Research*, 70, 56–74.
- Nourelfath, M., & Châtelet, E. (2012). Integrating production, inventory and maintenance planning for a parallel system with dependent components. *Reliability Engineering & System Safety*, 101, 59–66.
- Pan, Q. K., & Ruiz, R. (2013). A comprehensive review and evaluation of permutation flowshop heuristics to minimize flowtime. *Computers & Operations Research*, 40(1), 117–128.
- Park, M., & Kim, Y. (2000). A branch and bound algorithm for a production scheduling problem in an assembly system under due date constraints. *European Journal of Operational Research*, 123(3), 504–518.
- Pinedo, M. L. (2012). *Scheduling: Theory, algorithms, and systems*. Berlin: Springer.
- Rahim, M. A., & Ben-Daya, M. (2012). *Integrated models in production planning, inventory, quality, and maintenance*. Berlin: Springer.
- Rahmani, D., & Heydari, M. (2014). Robust and stable flow shop scheduling with unexpected arrivals of new jobs and uncertain processing times. *Journal of Manufacturing Systems*, 33(1), 84–92.
- Rasconi, R., Cesta, A., & Policella, N. (2010). Validating scheduling approaches against executional uncertainty. *Journal of Intelligent Manufacturing*, 21(1), 49–64.
- Reeves, C. (1995). A genetic algorithm for flowshop sequencing. *Computer and Operational Research*, 22, 5–13.
- Ribas, I., Leisten, R., & Framiñan, J. M. (2010). Review: Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. *Computers and Operations Research*, 37(8), 1439–1454.
- Roux, O., Duvivier, D., Quesnel, G., & Ramat, E. (2013). Optimization of preventive maintenance through a combined maintenance–production simulation model. *International Journal of Production Economics*, 143(1), 3–12.
- Rudek, A., & Rudek, R. (2013). Makespan minimization flowshop with position dependent job processing times computational complexity and solution algorithms. *Computers and Operations Research*, 40(8), 2071–2082.
- Ruiz, R., García-Díaz, J. C., & Maroto, C. (2007). Considering scheduling and preventive maintenance in the flowshop sequencing problem. *Computers and Operations Research*, 34(11), 3314–3330.
- Shafaei, R., & Brunn, P. (1999). Workshop scheduling using practical (inaccurate) data part 2: An investigation of the robustness of scheduling rules in a dynamic and stochastic environment. *International Journal of Production Research*, 37(18), 4105–4117.
- Shahidehpour, M., & Marwali, M. (2012). *Maintenance scheduling in restructured power systems*. Berlin: Springer.
- Shrouf, F., Ordieres-Meré, J., García-Sánchez, A., & Ortega-Mier, M. (2014). Optimizing the production scheduling of a single machine to minimize total energy consumption costs. *Journal of Cleaner Production*, 67, 197–207.
- Sloan, T. (2004). A periodic review production and maintenance model with random demand, deteriorating equipment, and binomial yield. *Journal of the Operational Research Society*, 55(6), 647–656.
- Sun, Y., Zhang, C., Gao, L., & Wang, X. (2011). Multi-objective optimization algorithms for flow shop scheduling problem: A review

- and prospects. *The International Journal of Advanced Manufacturing Technology*, 55(5–8), 723–739.
- Taillard, E. (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research* 64:278–285. <http://ina.eivd.ch/Collaborateurs/etd/default.htm>.
- Talbi, E. G. (2009). *Metaheuristics: From design to implementation* (Vol. 74). Hoboken: Wiley.
- Tambe, P. P., & Kulkarni, M. S. (2015). A superimposition based approach for maintenance and quality plan optimization with production schedule, availability, repair time and detection time constraints for a single machine. *Journal of Manufacturing Systems*, 37, 17–32.
- Tsai, Y., Wang, K., & HY, T. (2001). Optimizing preventive maintenance for mechanical components using genetic algorithms. *Reliability Engineering and System Safety*, 74, 89–97.
- Ventura, J. A., & Yoon, S. (2013). A new genetic algorithm for lot-streaming flow shop scheduling with limited capacity buffers. *Journal of Intelligent Manufacturing*, 24(6), 1185–1196.
- Vieira, G. E., Herrmann, J. W., & Lin, E. (2003). Rescheduling manufacturing systems: A framework of strategies, policies, and methods. *Journal of Scheduling*, 6(1), 39–62.
- Wang, S., & Liu, M. (2013). A branch and bound algorithm for single-machine production scheduling integrated with preventive maintenance planning. *International Journal of Production Research*, 51(3), 847–868.
- Weckman, G. A., Bondal, A., Rinder, M. M., & Young, W. A. I. I. (2012). Scheduling flexible job shop problem subject to machine breakdown with route changing and right-shift strategies. *Neural Computing and Applications*, 21(7), 1465–1475.
- Weinstein, L., & H, C. C. (1999). Integrated maintenance and production decisions in a hierarchical production planning environment. *Computer and Operations Research*, 26, 1059–1074.
- Wong, C. S., Chan, F. T. S., & Chung, S. H. (2013). A joint production scheduling approach considering multiple resources and preventive maintenance tasks. *International Journal of Production Research*, 51(3), 883–896.
- Xia, T., Jin, X., Xi, L., & Ni, J. (2015). Production-driven opportunistic maintenance for batch production based on mam-apb scheduling. *European Journal of Operational Research*, 240(3), 781–790.
- Xiong, J., Xing, L., & Chen, Y. (2013). Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns original research article. *International Journal of Production Economics*, 141(1), 112–126.
- Yagmahan, B., & Yenisey, M. M. (2009). Scheduling practice and recent developments in flow shop and job shop scheduling. *Studies in Computational Intelligence*, 230, 261–300.
- Yang, S. H., & Wang, J. B. (2011). Minimizing total weighted completion time in a two-machine flow shop scheduling under simple linear deterioration. *Applied Mathematics and Computation*, 217(9), 4819–4826.
- Yellig, E. J., & Mackulak, G. T. (1997). Robust deterministic scheduling in stochastic environments: The method of capacity hedge points. *International Journal of Production Research*, 35(2), 369–379.
- Yenisey, M. M., & Yagmahan, B. (2014). Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends. *Omega*, 45, 119–135.
- Zhao, C., & Tang, H. (2011). A note on two-machine no-wait flow shop scheduling with deteriorating jobs and machine availability constraints. *Optimization Letters*, 5(1), 183–190.