IPro-GA: an integrated prognostic based GA for scheduling jobs and predictive maintenance in a single multifunctional machine

A. Ladj [*] **C. Varnier** [**] **F. Benbouzid-Si Tayeb** [*]

[*] *Ecole nationale Supérieure d'Informatique of Algiers-Algeria (ESI), (e-mail: a_ladj@esi.dz , f_sitayeb@esi.dz).*
[**] *Institut Femto-St,Dept AS2M, Besançon France, (e-mail: christophe.varnier@ens2m.frs)*

Abstract: In highly competitive environment, manufacturing system availability has become a critical issue. For this reason, predictive maintenance must be properly integrated in the production scheduling in order to take into account the wear and tear of the equipment to prevent it from the failure risk. In this context, we investigate the problem of a single multifunctional machine subjected to predictive maintenance based on Prognostic Health Management (PHM). We propose a new interpretation of PHM outputs to define the machine degradation corresponding to the processing of every task. We design a genetic algorithm that we called IPro-GA with the objective of minimizing the total interventions cost. Computational results show the efficiency of our scheme with an average deviation of about 0.1% over a lower bound.

Keywords: production, predictive maintenance, Prognostic Health Management (PHM), joint maintenance and production scheduling, Genetic Algorithm (GA).

1. INTRODUCTION

As an important component of manufacturing industry, production scheduling has been extensively studied. To match the real world, this scheduling must take into account the unavailability of equipments due to breakdowns or maintenance operations. Thus, a maintenance planning has to be integrated in the production scheduling to balance resource availability and avoid conflictual situations. This problem is known in the literature as "production scheduling with availability constraints" or "integrated maintenance and production scheduling" (Hadidi *et al.,* 2012). This problems for different machine configurations, are strongly NP-hard, since for each separate criterion (production and maintenance) the problem is strongly NP-hard (Kubiak *et al.,* 2002). For this reason, exhaustive methods take a prohibitive execution time to find the best solution.

Works were previously proposed to investigate these problems. Two cases of consideration about the unavailability constraints can be found in the literature: (i) the deterministic case where intervals are known and fixed in advance and often correspond to preventive maintenance operations (Ma *et al.,* 2010), (ii) the dynamic case where unavailabilities periods are flexible and stand as decision variables. This is the case for instance when information about predictive maintenance are provided from a *Prognostic Health Management (PHM)* module.

For the latter strategy, prognostic is recognized as a key feature because it provides useful information to the maintenance decision process. It infers the current state and predicts the future progression of failure to estimate the time before a failure known as the *Remaining Useful Life* (*RUL*). In brief, it relies on the usage of condition monitoring (CM) data from operating equipment to obtain useful features, next assesses the level of degradation, and then predict the evolution of phenomena (Wang and Pecht 2011).

In this field, Ershun *et al.* (2012) proposed an integrated prognostics-based-scheduling model incorporating both production scheduling and predictive maintenance planning for a single machine with the objective of minimizing the maximum tardiness. Predictive maintenance operations are performed based on a new metric called Remaining Maintenance Life (RML), proposed to set a safety threshold before reaching the RUL. Considering wind farms systems, Kovacs *et al.* (2011) investigated the problem of optimizing the scheduling of maintenance actions. A mixed-integer programming (MIP) formulation is proposed to cover four categories of maintenance tasks so as to minimize the total production loss of the turbines. Varnier and Zerhouni (2012) proposed a mixed integer programming model to optimally solve scheduling production and predictive maintenance problem with the objective of minimizing the makespan and maintenance delays in flow-shop. In this study, machines are able to switch between two production modes: nominal and a sub-nominal one. The developed program allows finding both the best control mode for each machine and the best predictive maintenance policies. An interesting case of parallel machines was studied by Herr *et al.* (2014). Authors used PHM results to set a plateforme running

using different operating conditions. The main goal is to provide a prognostics-based schedule in order to reach a given demand as long as possible. A single predictive maintenance operation is considered. The objective is to extend the useful life of the whole platform. A second objective is to use machines' full potential.
The recent tremendous emergence of mechatronics offers exciting possibilities for the further evolution of machine tools. Due to this progress, various kinds of powerful single machine have been designed in the field of factory production (e.g., intelligent machine tool). Instead of using multiple specific-purpose machines, projects are nowadays working on multifunctional machinery (*Chameleon project* 2011).
Many researches dealing with single machine scheduling problem and investigating a vast set of problem specifications has been provided. Although most of these studies seek to optimize some measures, they do not usually match reality because they assume that machines are always available during the planning horizon. Therefore, the issue of integrating efficient maintenance strategy in production scheduling is becoming a prime necessity for failure-prone manufacturing systems (Wang and Liu, 2013).
With the advancements of sensor and intelligent prognostic technologies, deterministic maintenance, which induces excessive interventions and financial losses, is replaced by a more sophisticated strategy based on the estimation of RUL using a PHM system. In all previous studies, unique RUL value (expressed in unit of time) was estimated and used as a threshold to perform predictive maintenance operations independently of tasks being processed. In our paper, we propose a new interpretation of PHM results. We assume that a single multifunctional machine is subjected to many predictive maintenance interventions during the planning horizon. This equipment is supposed to be monitored continuously and a PHM module provides, due to different stress that induce various degradation level, for each kind of job the corresponding RUL. In our scheme, we introduce a new metric to express the degradation of the machine when processing each kind of job. In this context, we develop an integrated prognostic based genetic algorithm IPro-GA for scheduling production and predictive maintenance with the objective of minimizing the total maintenance cost.
The remaining content of the paper is organized as follows. In section 2, we present the scheduling problem. In section 3, the proposed genetic algorithm is developed as well as integrated genetic operators. Finally, experiments results of the newly designed GA are discussed. A general conclusion of the work and the perspectives considered are given in the last section.

2. PROBLEM STATEMENT

We have to schedule simultaneously a set $\mathcal{J}$ of n production tasks being processed by a machine subjected to predictive maintenance interventions. Thus, the resulted joint scheduling integrates both production and maintenance activities. This scheduling can be seen as a succession of several task batches separated by predictive maintenance operation, denoted by $\pi = \{\mathcal{B}_1, \mathcal{M}_1, \ldots, \mathcal{M}_{l-1}, \mathcal{B}_l\}$, where $\mathcal{B}_i$ is the i^{th} block of jobs, $\mathcal{M}_i$ is the i^{th} maintenance activity, l is the minimum number of bocks required to process all jobs $\bigcup_{i=1}^{n} \mathcal{B}_i = \mathcal{J}$. Each job J_i is included strictly in one production block $\forall i, j \in \{1, \ldots, n\}, i \neq j : \mathcal{B}_i \bigcap \mathcal{B}_j = \varnothing$.
The problem consists then in determining the jobs assignment for each block to minimize the total cost required to process all maintenance operations. The job sequence must be well arranged in order to make full use of the machine. In this section, we remind first the single machine production scheduling problem, and then we define both PHM problem and predictive maintenance problem.

2.1 Production scheduling problem

In this paper, we address the problem of scheduling a set of n independent non-resumable jobs $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$ on a single machine. There are several assumptions that are commonly made regarding this problem (Merten and Muller, 1972):

- All the jobs are available at time zero;
- Each job requires a given known, deterministic and non negative processing time, denoted p_i;
- Machine is not continuously available due to predictive maintenance operations. When available, it can only process one job at a time.

Hence, the production scheduling problem is how to effectively arrange the sequence of tasks to be performed by the single machine.

2.2 Prognostic Health Management problem

It is assumed that the machine is monitored continuously by a PHM module. Sensors provide the most representative information of machine degradation. As diverse tasks are processed on single multifunctional machine, this latter is subjected to a deterioration process that depends on the job being executed because every kind of job requires specific functionalities that cause various levels of damage for the equipment. Hence, under these conditions, each job J_i has an associated remaining useful life value RUL_i.
We consider the following assumptions:

- A deteriorating prognosis system provides RUL_i of the machine corresponding to a given Job J_i;
- The resulting RUL is assumed certain;
- We consider that no accidental failure can occurred during the schedule horizon.

Using the PHM outputs, a degradation value δ_i is calculated for every job J_i. $\delta_i \in]0;1[$ represents the wear and tear of the machine when only job J_i is processed during the processing time p_i (0 means no degradation committed, 1 a degradation of 100%). If we consider a " as good as new machine" , RUL_i is then the period during which we could produce job J_i before a machine failure. Hence, $\delta_i = f(p_i)$ where f characterizes the evolution of the machine degradation. We consider in our model that f could be a linear function. For each job J_i, $\delta_i = \alpha * p_i$ where $\alpha = \frac{1}{RUL_i}$, so $\delta_i = \frac{p_i}{RUL_i}$ as shown in Fig. 1.

2.3 Predictive maintenance scheduling problem

Predictive maintenance operations reduce the risk of machine failures and restore the machine to "as good as new" state. Let Δ be the maximal authorized degradation of

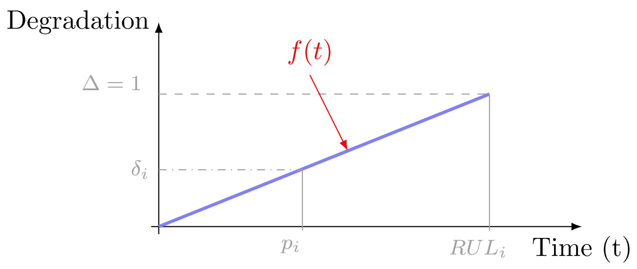

Fig. 1. Illustration of the relationship between p_i, RUL_i and δ_i

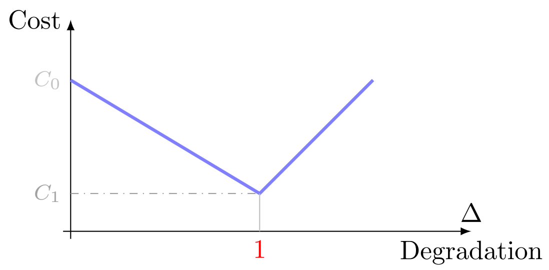

Fig. 2. Predictive maintenance cost model

machine. Beyond this threshold, a predictive maintenance task should be planned. The accumulated degradation of a set of jobs processed between two consecutive maintenance operations must be smaller than this maximal value $\sum_{J_i \in Block} \delta_i \leqslant 1$. In our case, we fix $\Delta = 1$. Considering that, in general, the machine could not process all the jobs before maintenance operation and then more than one production batch should be programmed. The maintenance cost, is divided in two (2) parts : a fixed cost and a variable one (Fig. 2):

- C_f: represents the optimal maintenance cost spent when the machine reached a full degradation Δ ;
- $C_v(\Delta)$: is a function of the level of machine degradation reached at the maintenance time.

We consider the following assumptions:
- After a predictive maintenance operation, the machine is recovered to be "as good as new", i.e. the machine is renewed and its accumulated degradation is set to 0.
- During the planning horizon, at least one predictive maintenance operation is performed: $\sum_{i=1}^{n} \delta_i > \Delta$.
- No maintenance operation is performed after the processing of last block of tasks.

3. IPRO-GA: INTEGRATED PROGNOSTIC BASED GENETIC ALGORITHM

As introduced above, the problem studied is to create a prognostic based joint scheduling of several jobs on a single multifunctional machine with the objective of minimizing the total maintenance cost. The planning horizon can be divided on multiple cycles of production separated by predictive maintenance interventions. First, machine's RUL is estimated by the PHM module for each kind of production. Then the degradation values corresponding to each job are obtained. Next, with the obtained information, the first block of tasks to be performed is generated respecting the predetermined constraint of machine maximal capacity Δ. At the end of the first block, a predictive maintenance is scheduled to recover the machine to a "as good as new" state. The cost of this latter intervention is calculated based on the accumulated degradation of jobs included in the first block. After that, given the rest of jobs, a new block will be determined and launched and the new maintenance cost is added to the total one. The same procedure will be iterated till all jobs are scheduled.
An important factor in the studied problem is the integration of maintenance information based on PHM outputs. To seek this goal, a new GA is designed. IPro-GA lays emphasis on the best interpretation of PHM results. The next subsections will describe the different genetic operators incorporated to our scheme.

3.1 Solutions encoding and decoding

The representation step specifies the mapping from the individuals (candidate solutions) into a set of genotypes. In our GA, a genotype is expressed by sequencing the job sets for all the production batches. A set of jobs numbers in one batch corresponds to a gene. For example, for an instance of problem $\mathcal{J} = \{J_1, J_2, \ldots, J_{10}\}$, a candidate solution is $\pi = \{(5, 10)(2, 9, 1, 3)(6, 7, 8)(4)\}$. We decode this representation by scheduling the jobs of the first block (J_5 and J_{10}), then performing the first predictive maintenance operation with a cost depending on assigned jobs. Next, we iterate the same process for the rest of the blocks one by one.

3.2 Fitness function

GA needs a fitness function to evaluate the quality of an individual in the population. For our problem, giving a candidate solution $\pi = \{\mathcal{B}_1, \mathcal{M}_1, \ldots, \mathcal{M}_{l-1}, \mathcal{B}_l\}$, the total predictive maintenance cost is calculated from Fig. 2:

$$Cost^{PM}(\pi) = \sum_{k=1}^{l-1} (C_0 + (C_1 - C_0) \sum_{J_i \in \mathcal{B}_k} \delta_i) \qquad (1)$$

Therefore, the reciprocal of the total maintenance cost is selected as the affinity function for our GA:
$Affinity(\pi) = \frac{1}{Cost^{PM}(\pi)}$. It can be noticed that the lower the predictive maintenance cost is, the higher the affinity value is and so the better the solution is.

3.3 Population initialization

Instead of starting with an initial population randomly generated, it seems more efficient to use special techniques to produce a higher quality initial population (Reeves 1995). Initial population of $PopSize$ individuals is generated as follow:
- The first part of the initial population ($\alpha\% PopSize$) is generated by a uniform distribution. Its purpose is to ensure diversity of the research. First, a random permutation of all jobs is generated. Then First Fit heuristic (Coffman *et al.*, 1984) is applied to form a candidate solution.
- The remaining part ($(100 - \alpha)\% PopSize$) is generated using the two common heuristics First Fit Decreasing (FFD) and Best Fit Decreasing (BFD) heuristics (Coffman *et al.*, 1984). It exploits the characteristics of these good

solution to form other solutions by applying a series of permutations between jobs. In this way, we ensure that this part of the population is formed by fit members.

3.4 Selection operator

For the sake of simplicity, we have chosen classical selection scheme, 2-tournament selection (Michalewicz 1996). It consists on randomly choosing two members from the current population and selects the fittest one.

3.5 Crossover operator

Since the considered objective is the minimizing of maintenance cost, by analyzing the cost evolution model we deduct that ideally, a preventive maintenance operation is planned when the machine degradation value reaches the maximal threshold Δ. Thus, it is clear that we should make *full* use of the machine and then build as full as possible production bins. To seek this goal, we modified the crossover operator used by Rohlfshagen and Bullinaria (2010). This crossover was applied to a Bin Packing problem and produced a single offspring by copying the fullest bins from parents.
In our case, with a probability equals to $CrossProb$, the proposed crossover operator produces two offspring. In the first phase, starting from an empty offspring, we copy the fullest non-overlapping blocks from parents. In other words, blocks from both parents are sorted in the order of non-increasing size (degradation) and then each block is copied in offspring only if it contains no duplicated job. In the second phase, we need to represent the parents as job sequence (permutation of jobs). We scan the sequence of the first parent, respectively the second, from left to right, skipping jobs that are already contained and assign the rest jobs to the first offspring, respectively the second, using the First Fit rule. Our crossover naturally avoids generating infeasible solutions, where the capacity constraint is violated because the sum of jobs degradation in a block exceeds the maximum capacity Δ. This can be explained by the use of FF heuristic. This eliminates the time that would be spent cutting down infeasibilities.

3.6 Mutation operator

We chose a simple mutation method inspired from the classical SWAP mutation (Michalewicz 1996). It consists on randomly swapping, if possible, two selected jobs from two different blocks. We only allow mutations that guarantee the feasibility of the obtained solutions. Thus, the maximal capacity Δ must be respected for each block. The mutation probability is set to $MutProb$. The number of permutation is randomly fixed between a low and a high limit.

3.7 Replacement

Individuals of the next generation are selected from the whole population formed by parents and newly created children. β% from the worst individuals are directly inserted in the new population. Then, we complete the rest of the population by the fittest members from parents and children. However, a common problem is that the population could sometimes stall around a local optimum. To avoid that, we apply a restart mechanism in our GA based on the scheme used by Ruiz *et al.* (2006). If the best solution found doesnt improve after $MaxImprovGen$ generations, we should update the current population so as to improve its diversity. The best γ% of the population are skipped, 50% from the remaining individuals suffer a mutation and the rest are replaced by newly random created members using the First Fit ordering.

3.8 Stopping criteria

In traditional GA, either computation time or the number of generations is selected as termination criterion. Our algorithm terminates after $MaxGen$ generations.

4. COMPUTATIONAL RESULTS

In this section, we present the results of series of computational experiments, conducted to test the newly proposed GA. They were tested on a PC with Intel® Core™ i3-2330M CPU @ 2.20 GHz and 2.00 Go RAM. The heuristic algorithms are coded in $C++$.

4.1 Data generation

We generate a variety of random testing instances where:
- Size of problem instances $n \in [20, 300]$;
- Processing time of jobs is selected from a uniform distribution $p_i \in U[1, 50]$;
- RUL of each job is selected from a uniform distribution $RUL_i \in U[100, 150]$ for $n \in [20, 100]$, $RUL_i \in U[100, 200]$ for $n \in]100, 200]$, $RUL_i \in U[100, 250]$ for $n \in]200, 300]$;
- Maintenance costs are set $C_1 = 100$, $C_0 = 1000$.

10 instances are generated and tested for each problem size. We run 10 independent replicates of each instance in order to have a better picture of the results. We average the results for all the given instances.
We proceed with the analysis of the results of pilot experiments generated randomly from a factorial design, and then we fix every factor to the most interesting level. We set: $PopSize = 200$, α =85, $\beta = 20$, $\gamma = 25$, $MaxGen = 300$, and $MaxImprovGen = 20$. For operators probabilities: $CrossProb = 0.7$, and $MutProb = 0.015$.

4.2 Performance analysis of the proposed GA

In Table 1, we can see a comparison of the execution time CPU (in *ms*), the total cost $Cost^{PM}$ between our GA and the best known heuristic BFD (Coffman *et al.*, 1984), and then percentage of cost reduction is calculated ($Cost^{PM}\%\searrow$). These results are the average of 10 instances for each problem size. It is clear that our GA is the slowest to solve problem instances. Its execution time is much more greater than BFD heuristic since it manipulates a large set of individuals on which it applies greedy genetic operators during the whole process. In the other hand, there is a significant difference between the cost obtained by our GA and by BFD. The proposed GA offers better solutions in all cases.

4.3 Lower bound

Since no optimal solutions are known for the studied problem, we compare our GA results against a lower bound,

Table 1. Comparison of CPU and $Cost^{PM}$ between BFD and $IPro-GA$

n	BFD		$IPro-GA$		$Cost^{PM}$
	$CPU(ms)$	$Cost^{PM}$	$CPU(ms)$	$Cost^{PM}$	$\% \searrow$
20	0.1026	317.56	2509	270.17	14.92
40	0.2673	713.11	3840	650.23	8.82
60	0.3193	1052.97	5732	1021.11	3.03
80	0.6208	1456.49	6584	1402.53	3.70
100	0.6620	1750.42	7118	1692.47	3.31
120	0.9759	2130.01	8160	2072.06	2.72
140	0.9775	2489.72	7226	2443.98	1.84
160	0.9882	2713.33	6746	2662.57	1.87
180	1.1573	3134.46	7323	3083.03	1.64
200	1.7568	3547.92	7545	3493.15	1.54
250	2.8641	4419.74	7726	4354.08	1.49
300	4.0524	5726.34	7754	5615.45	1.94

denoted $Cost_{low}^{PM}$. We note l_{low} the smallest number of blocks of capacity Δ required to process all jobs. In other words, if we suppose that all job blocks are completely full, i.e. their degradation is equal to Δ, then $l_{low} = \lceil \frac{1}{\Delta} \sum_{i=1}^{n} \delta_i \rceil$, and thus the total maintenance cost for all production batch, except the last one, will be fixed to cost C_1. Then, the low bound can be estimated as $Cost_{low}^{PM} = (l_{low}-1)C_1$. Table 2 shows the comparison between the maintenance cost $Cost^{PM}$ generated by our GA and the lower bound $Cost_{low}^{PM}$. One can easily observe that IPro-GA yields a very small deviation from the lower bound. In worst cases, predictive maintenance costs increase by less than 0.2%. Moreover, in several cases, this deviation is less than 0.1%. That confirms the efficiency of our GA to generate best solutions for all problem instances. This is argued by the correct parameters setting and the choice of appropriate genetic operators, especially the crossover that preserves good characteristics of parents and transfers them to offspring.

5. CONCLUSION

In this paper we have proposed a new genetic algorithm IPro-GA to solve the integrated production and predictive maintenance scheduling on a single machine under the total cost minimization criterion. Since each kind of production requires specific machine functionalities, we have assumed that a PHM system provides the RUL corresponding to each kind of production and then a relative degradation value is calculated. A predictive intervention is scheduled whenever the maximal authorized threshold is reached. The proposed algorithm include carefully designed operators in order to enhance the quality of the obtained solutions. We have conducted various experiments that showed the efficiency of the proposed algorithm.
Further topics would be continued with regards to this results. The proposed integrated scheduling model can be extended to manage other typologies of production systems. Another work can deal with the multi-objective character of the investigated optimization problem.

Table 2. Comparison of $Cost^{PM}$ between the $LowerBound$ and $IPro-GA$

n	$IPro-GA$	$Cost_{low}^{PM}$	$\% \nearrow$
	$Cost^{PM}$		$Cost^{PM}$
20	270.1702	270	0.0630
40	650.2275	650	0.0350
60	1021.105	1020	0.1083
80	1402.533	1400	0.1809
100	1692.468	1690	0.1460
120	2072.060	2070	0.0995
140	2443.977	2440	0.1629
160	2662.571	2660	0.0966
180	3083.025	3080	0.0982
200	3493.147	3490	0.0902
250	4354.084	4350	0.0939
300	5615.446	5610	0.0971

REFERENCES

Alcaraz, J., Maroto, C. (2001), A robust genetic algorithm for resource allocation in project scheduling. *Annals of Operations Research*, volume 102(1),83109.

Chameleon project (2011). Final Report-CHAMELEON (Production dependent adaptive machine tool). European Commission (EC)'s Seventh Framework Programme (FP7), Available from: ¡http://www.chameleonproject.eu¿. [28 September 2015].

Coffman, E.G., Carey, M.R. and Johnson, D.S. (1984), Approximation Algorithms for Binpacking - An updated Survey, A*lgorithm Design for Computer System Design, Springer.*

Ershun, P. , Wenzhu, L., Lifeng, X. (2012), A joint model of production scheduling and predictive maintenance for minimizing job tardiness, *The International Journal of Advanced Manufacturing Technology*, volume 60(9), 1049-1061.

Goldherg, D.E. (1989), Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley.

Hadidi, L.A., Al-Turki, U.M., and Rahim, A. (2012), Integrated models in production planning and scheduling, maintenance and quality: a review. *International Journal of Industrial and Systems Engineering*, volume 10(1), 21-50.

Herr, N., Nicod, J.M., Varnier, C. (2014). Prognostics-based scheduling in a distributed platform: Model, complexity and resolution. IEEE International Conference on Automation Science and Engineering, 1054-1059.

Kovacs, A., Erdos, G., Monostori, L., and Viharos, Z. J. (2011), Scheduling the maintenance of wind farms for minimizing production loss. *Proc. of the 18th IFAC World Congress*, Milano, Italy.

Kubiak, W., Blazewicz, J., Formanowicz, P., Breit, J., and Schmidt, G. (2002) Two-machine flow shops with limited machine availability. *Europeen Journal of Operational Researchs*, volume 136, 528-540.

Ma, Y., Chu, C., Zuo, C. (2010). A survey of scheduling with deterministic machine availability constraints. *Computers & Industrial Engineering*, volume 2(58), 199-211.

Merten, A.G., and Muller, M.E. (1972) Variance minimization in single machine sequencing problems. Manag Sci, volume 18,518-528.

Michalewicz, Z. (1996), Genetic algorithms + data structures = evolution programs. 3rd ed., Heidelberg: Springer, Berlin.

Reeves, R.C., (1995).A genitic algorithm for flowshop sequencing Department of Statistics and Operational Research- Coventry University, England.

Rohlfshagen, P., Bullinaria, J.A., (2010). Nature inspired genetic algorithms for hard packing problems, *Annals of Operations Research* , volume 179(1),393-419.

Ruiz, R., Maroto; C., Alcaraz, J., (2006), Two new robust genetic algorithms for the flowshop scheduling problem. *Omega* volume 34(5), 461-47.

Varnier, C., Zerhouni, N., (2012). Scheduling predictive maintenance in flow-shop. IEEE Conference on Prognostics and System Health Management, PHM'12., China.

Wang, S., and Liu, M., (2013). A branch and bound algorithm for single-machine production scheduling integrated with preventive maintenance planning, *International Journal of Production Research*, volume 51, 847-868.

Wang, W., and Pecht, M. (2011). Economic analysis of canary-based prognostics and health management. *IEEE Trans. Ind. Electron.*, volume 58(7), 3077-3089.