Optimization for a joint predictive maintenance and job scheduling problem in manufacturing industry

Abstract

While job scheduling problems have been studied extensively, scheduling problems with endogenous yield rates that may be affected by predictive maintenance is seldom investigated. In practice, a decision maker may find some time to insert predictive maintenance onto some machines. This occupies machines and delay the processing of jobs but increases yield rates to shorten future job processing times. In this study, we consider the optimization a joint predictive maintenance and job scheduling problem for the minimization of total tardiness. We formulate a mixed integer program for this problem and develop a heuristic algorithm based on Tabu search. We demonstrate the effectiveness of our algorithm through numerical experiments.

Keywords: Predictive maintenance, job scheduling, yield rate, mixed integer programming, Tabu search.

1 Introduction

Maintenance is an integral component of operating manufacturing equipment. There are a lot of past studies in different maintenance models and policies (Wang, 2002). Most types of maintenance fall under two main categories: preventive and corrective maintenance (Mobley, 2002). *Corrective maintenance*, also known as run-to-failure maintenance, is a reactive management technique which is initiated only when a machine breaks down. On the other hand, *preventive maintenance* is based on elapsed time or hours of operation. *Predictive maintenance* is one kind of condition-driven preventive maintenance, which determines the maintenance schedule of each machine by monitoring the mechanical condition, system efficiency, and other indicators. In contrast to the traditional preventive maintenance, which is only conducted on the same schedule every cycle, predictive maintenance is conducted as needed, drawing on real-time collection and analysis of machine operation data to identify issues at the nascent stage before they may interrupt production. Predictive maintenance provides a flexible way to determine maintenance schedule and often involves prognostic and health management (PHM), which is more complex and may require new equipment and technology in order to collect and share data with a centralized system.

Taiwan is one of the world's largest providers in the electronics manufacturing industry. The production process in the electronics manufacturing industry strongly relies on the availability of machines and usually spends a high cost on machine maintenance. Although machines nowaday are more advanced and reliable than those in the past, they are still subject to deterioration because of ageing and requires maintenance. The planned production schedule is thus often influenced consequently. Therefore, maintenance plays an important role in production scheduling problems, especially in the electronics manufacturing industry.

However, since predictive maintenance is usually complex and the maintenance effect is not well predicted, maintenance and production issues are usually considered separately. Most of the studies do not simultaneously take scheduling and maintenance planning decisions into account, which means the production quality may be overestimated and the production cost may be underestimated. We propose a general model which considers both

production scheduling and maintenance planning decisions, and the objective is to minimize the total cost.

In our study, we consider a job scheduling problem with multiple parallel machines, all with yield rates declining in time. There are jobs with different demand quantities ready to be processed. Each job must be processed from a machine to fulfill certain demand quantities. A decision maker will decide the production and maintenance schedule for each machine. The maintenance schedule will affect the yield rates of machines. More precisely, after a certain amount of time is spent on maintenance, a machine's yield rate will be increased. However, that time investment in maintenance will force some jobs to be delayed, which may or may not be a right decision. If the total production quantity of a job does not meet the demand, there will be a shortage penalty included in the total cost. The problem is formulated as a mixed integer linear program in which the objective function is to minimize the total cost with maintenance and production constraints. As our problem is NP-hard, it is believed that there is no polynomial time algorithm for solving general scheduling problems. Therefore, we propose a heuristic algorithm based on Tabu search to find a near-optimal solution in acceptable time.

2 Literature Review

Aghezzaf, Jamali, and Ait-Kadi (2007) propose a joint production and maintenance planning model for a production system , where production capacity is reduced when preventive or corrective maintenance activities happen. Aghezzaf and Najid (2008) extend the forward formulation with multi-line production systems subject to random failures, and they also present a Lagrangian-based heuristic procedure for the solution of the initial planning model. Najid, Alaoui-Selsouli, and Mohafid (2011) present a model where preventive maintenance tasks are carried out in time windows to better meet customer demand. The model also allows shortage and considers the tradeoff of shortage cost and inventory cost. The idea of combining preventive maintenance and random breakdowns into the economic manufacturing quantity (EMQ) model has been widely concerned. Rezg, Xie, and Mati (2004) present three

models with different production control policies. The paper also present a simulation model by using genetic algorithms.

The issue of unreliable production systems has been widely concerned at the scheduling level. There is a stream of literature considering a known maintenance schedule (Lee, 1996; Schmidt, 2000; Ma et al., 2010), and these scheduling problems are reduced into pure scheduling problems with machine availability constraints.

Lee and Chen (2000) study the problem of scheduling jobs and one time of maintenance on parallel machines where each machine must be maintained once during the planning horizon. The objective is to minimize the total weighted completion time. The research typically considers only one maintenance during the whole planning horizon. However, there is a stream of papers dealing with finding a periodic schedule for fixed-length maintenance periods. Cassady and Kutanoglu (2005) examine the problem for finding the optimal joint production schedule and a periodic preventive maintenance schedule to minimize the total tardiness. Kubzin and Strusevich (2006) propose a multistage scheduling systems, the flow shop and the open shop, in which the processing of a job requires several operations to be performed consecutively. The objective is to minimize the total makespan since that the maintenance periods are mandatory, and the decision maker has to schedule them along with the jobs to be processed.

3 Problem description and formulation

We consider a single product job scheduling problem with maintenance. There are jobs with different demand levels, due time, and weight required to be satisfied. A decision-maker needs to decide both the production schedule and the maintenance schedule for each machine to minimize the total weighted penalty for the shortage of tardy jobs. The production schedule includes jobs dispatching and sequencing. The maintenance schedule decides when to maintain the machine.

We are given a planning horizon $T = \{1, 2, ..., n_T\}$ including n_T periods of fixed length. Let $N = \{1, 2, ..., n_N\}$ denote the set of jobs we need to fulfill and $M = \{1, 2, ..., n_M\}$

represent the set of machines. Each job requires a certain demand quantity Q_j at its given due time D_j. We also allow shortage in this problem and give each job a weighted penalty W_j when a unit of shortage happens.

On the other hand, each machine has different production efficiency, and we denote A_i as the ideal production rate of machine i. Since the yield rate of each machine decreases as time progresses, we denote r_{it} as the yield rate of the machine i at time t. The yield rate declines as a function of time. We formulate the function as a linear function with a constant decline rate to describe the deteriorating production system. Let B_i denote the yield declining rate of machine i, which means that the yield rate of machine i decreases B_i units every time. Both the yield rate and yield declining rate are in the range of 0 to 1. Here is a simple example. Given that machine 1 produces 100 units of products per time period, and that the current yield rate and the yield declining rate are 90% and 5% respectively, we may derive that $A_1 = 100$, $r_{1,t} = 0.9$, $B_1 = 0.05$, and the yield rate at next time period $r_{1,t+1}$ is $0.9 - 0.05 = 0.85$. The production quantity of the machine i in time period $t + 1$ is $100 \times 0.85 = 85$ units.

In the real world, the yield rate typically does not drop to zero and usually has a lower bound for each machine, we denoted the lower bound as L_i for machine i. Without maintenance, the yield rate would keep declining until it reaches L_i. As the yield rate falls at L_i, it remains constant without maintenance. There is a limitation on the total number of machines under maintenance within a time. We may arrange maintenance on any machine at any time as long as the number of machines under maintenance at the time does not exceed H. Machine i consumes F_i units of consecutive periods to take maintenance. Each maintenance process does not allow to be suspended and the production of the machine must be ceased during the maintenance. The yield rate of machine i will recover and rise to 1 immediately at the next time of the completion of its maintenance, and the yield rate will keep declining at a constant rate of B_i from 1 later on.

A job is required to be processed by only one machine with no preemption. Once a machine starts to process a job, it consumes several consecutive periods to accumulate pro-

duction quantity for the job. The difference between the demand quantity and production quantity accumulated by the due time is the shortage amount. Note that once a job is started on a machine, no other job or maintenance can be started until the machine stops processing the current job.

The decision maker decides how to dispatch and sequence the jobs, and also when to book the maintenance. To model this, let $y_{ijt} \in \{0, 1\}$ be 1 if machine i is processing job j in time period t or 0 otherwise. If machine i is processing job j in time period t, there will be a positive production amount g_{ijt}. Since we are allowed to assign each job to only one machine, we set $z_{ij} \in \{0, 1\}$ as 1 if job j is processed by machine i or 0 otherwise. To consider the maintenance schedule, let $x_{it} \in \{0, 1\}$ represent whether machine i is under maintenance in time t ($x_{it} = 1$) or not ($x_{it} = 0$). To ensure that the yield rate is no lower than the lower bound L_i, a binary variable $s_{it} \in \{0, 1\}$ is added to label whether the declined yield rate without adjustment will be lower than L_i. More precisely, s_{it} is 1 if $r_{i,t-1} - B_i > L_i$ or 0 otherwise. The last decision variables $e_{it} \in \{0, 1\}$ is 1 if time t is exactly the time right after maintenance or 0 otherwise.

The complete formulation is

$$\min \quad \sum_{j\in J}(Q_j - \sum_{i\in M}\sum_{t=1}^{D_j} g_{ijt})W_j \tag{1}$$

$$\text{s.t.} \quad \sum_{t\in T} g_{ijt} \le Q_j z_{ij} \qquad \forall i \in M, \quad \forall j \in N \tag{2}$$

$$\sum_{i\in M} z_{ij} = 1 \qquad \forall j \in N \tag{3}$$

$$g_{ijt} \le A_i y_{ijt} \qquad \forall i \in M, \quad \forall j \in N, \quad \forall t \in T \tag{4}$$

$$\sum_{j\in N} y_{ijt} \le 1 \qquad \forall i \in M, \quad \forall t \in T \tag{5}$$

$$\sum_{j\in N} g_{ijt} \le A_i r_{it} \qquad \forall i \in M, \quad \forall t \in T \tag{6}$$

$$\sum_{j\in N} g_{ijt} \le A_i(1 - x_{it}) \qquad \forall i \in M, \quad \forall t \in T \tag{7}$$

$$\sum_{i\in M} x_{it} \le H \qquad \forall t \in T \tag{8}$$

$$\sum_{t\in T}\sum_{i\in M} g_{ijt} \ge Q_j \qquad \forall j \in N \tag{9}$$

$$2e_{it} \le x_{i,t-1} - x_{it} + 1 \qquad \forall i \in M, \quad \forall t \in T \tag{10}$$

$$r_{it} \le Ke_{it} + Ks_{it} + r_{i,t-1} - B_i \qquad \forall i \in M, \quad \forall t \in T \tag{11}$$

$$r_{it} \le Ke_{it} + K(1 - s_{it}) + L_i \qquad \forall i \in M, \quad \forall t \in T \tag{12}$$

$$r_{it} \le 1 \qquad \forall i \in M, \quad \forall t \in T \tag{13}$$

$$F_i - \sum_{k=t-F_i}^{t-1} x_{ik} \le K(1 - e_{it}) \qquad \forall i \in M, \quad \forall t = F_i + 1, \cdots, |T| \tag{14}$$

$$x_{i,0} = 0 \qquad \forall i \in M \tag{15}$$

$$r_{it} \ge L_i; x_{it}, s_{it}, e_{it} \in \{0, 1\} \qquad \forall i \in M, \quad \forall t \in T \tag{16}$$

$$g_{ijt} \ge 0; y_{ijt} \in \{0, 1\} \qquad \forall i \in M, \quad \forall j \in N, \quad \forall t \in T \tag{17}$$

$$z_{ij} \in \{0, 1\} \qquad \forall i \in M, \quad \forall j \in N \tag{18}$$

where K is a large enough number.

The objective (1) is to minimize the total weighted penalty for the shortage of tardy jobs

in a scheduling problem. Constraints (2) define that the total production for job j will not higher than the demand quantity. Constraints (3) ensure that each job can only be assigned to one machine. Constraints (4) and (5) ensure that each machine can only deal with one job at a time. Constraints (6) define the production rate after considering the yield of each machine in every period. Constraints (7) ensure that each machine should stop producing during maintenance. Constraints (8) define the upper bound of the number of machines under maintenance in the same period. Constraints (9) ensure that each job should be fulfilled. Constraints (10) to (13) define that the yield of each machine would recover and rise to 1 immediately at the next period of the completion of its maintenance; otherwise, the yield would decrease, but it should never lower than the lower bound. Constraints (14) ensure that the maintenance for each machine should be continuous. Constraints (15) state that the machine would never be maintained at the dummy time period 0. Constraints (16) to (18) set the lower bound of r_{it} as L_i, that of g_{ijt} as 0, and require x_{it}, y_{ijt}, z_{ij}, s_{it}, and e_{it} to be binary.

Tables 1 and 2 introduce all the notations mentioned above.

Notation	Definition
x_{it}	1 if machine i is under maintenance at the time point t or 0 otherwise.
y_{ijt}	1 if machine i is processing job j in time period t or 0 otherwise.
z_{ij}	1 if machine i processes job j or 0 otherwise.
g_{ijt}	Production quantity of job j on machine i in time period t or 0 otherwise.
r_{it}	Yield rate of machine i in time period t, $r_{it} \in [0, 1]$.
s_{it}	1 $r_{i,t-1} - B_i < L_i$ or 0 otherwise.
e_{it}	1 if machine i finishes maintenance in time period $t-1$ or 0 otherwise.

Table 1: List of variables

Notation	Definition
M	set of machines.
N	set of jobs.
T	set of periods.
n_M	number machines.
n_N	number of jobs.
n_T	number of periods.
A_i	Ideal production rate of machine i.
B_i	Yield declining rate of machine i, $B_i \in [0, 1]$.
F_i	Number of consecutive time periods to complete maintenance for machine i.
L_i	Minimum yield rate of machine i, $L_i \in [0, 1]$.
Q_j	Demand quantity of job j.
D_j	Due time of job j.
H	Maximum number of machines that may be under maintenance in a period.

Table 2: List of sets and parameters

4 Algorithm

Our algorithm can be divided into three parts. The first part is a listing procedure which dispatch jobs to every machine by some rules. This listing approach will generate a initial feasible solution with no maintenance. We then use a greedy approach to find the best maintenance schedule for the initial solution. To improve this initial listing solution, we design a Tabu search algorithm and iteratively find better solutions by exchanging neighboring jobs until certain stopping criteria are met. The best solution which has the minimize total weighted penalties will be the final Tabu solution for this problem. The overall algorithm process is illustrated in Figure 1.

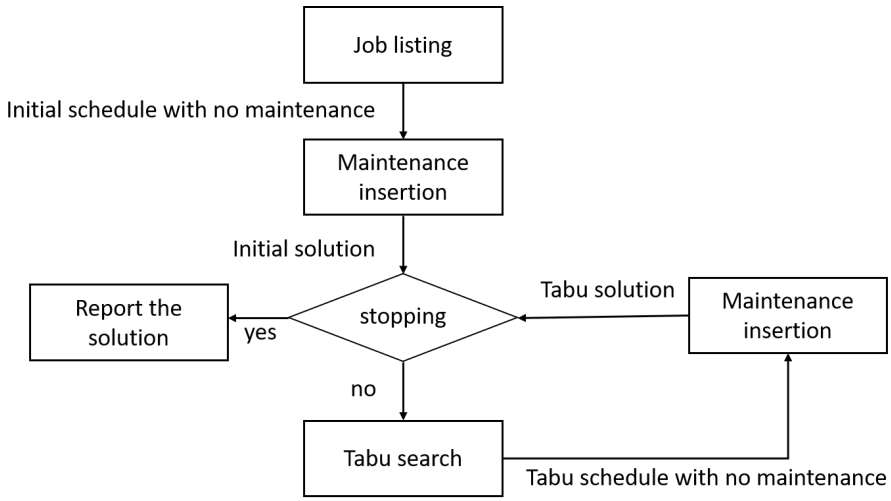

Figure 1: The overall process of the algorithm

4.1 Job listing

The first part in our algorithm is a listing approach. To assign jobs to machines, we consider three different listing rules.

The first rule is to list the jobs by the sequence of the largest penalty W_j first, the earliest due date/time (EDD) D_j first, and the least demand quantity D_j first (similar to the idea of shortest processing time used in the scheduling literature). The second rule is to calculate a ratio for each job and choose the job with the smallest ratio first. The ratio function is defined as

$$\frac{\text{the due time of the job}}{\text{the weighted penalty of the job}} = \frac{D_j}{W_j}. \tag{19}$$

If two ratios are equal,we choose the job with the smaller demand quantity. For both listing rules above, we assign jobs one by one to the machine which has the smallest cumulated work time after listing the jobs by rules.

The last rule is to calculate a ratio for each unscheduled job whenever a machine becomes available and assign the job with the smallest ratio to the available machine. Before choosing the jobs to dispatch, we first decide the machine which has the smallest cumulated work time t, which means the job we assign to the machine will start at time t. We then calculate the

ratio for each job as

$$\frac{\text{the time length between } t \text{ and the due time}}{\text{the weighted penalty of the job}} = \frac{D_j - t}{W_j}. \tag{20}$$

If two ratios are equal,we choose the job with the smaller demand quantity.

4.2 A greedy maintenance insertion procedure

The first algorithm above completes assigning all the jobs to each machine, and the next step is to decide the maintenance timing between jobs by the second algorithm. The second algorithm is a greedy algorithm. We iteratively try all the possible periods on all the machines to insert a maintenance schedule and choose the best one. For example, if there are n_N jobs, there will be totally n_N possible periods of time to decide whether to maintain or not before the job started. We repeat this process until no more maintenance schedule can improve the objective values. The maintenance schedule should also consider the max number of maintenance machines at the same time H due to the limited resources. The maintenance time of machine i should consumes F_i units of consecutive periods. If the number of maintenance at a given timing has reached the maximum possible number, we skip that maintenance. The listing algorithms and the greedy algorithm we mention above will generate an initial feasible solution with maintenance.

4.3 Job swapping based on Tabu search

The last algorithm is a Tabu search, in which we design a method to randomly exchange the neighboring jobs in the solution and update the best solution if the new solution has lower tardiness. The neighboring jobs mean all the jobs in the same order on each machine. In other words, the first job at each machine will be regarded as order 1, the second job at each machine will be regarded as order 2, and so on. Take Figure 2 as an example. Jobs 1, 5, and 6 will be regarded as the neighboring jobs in order 1. Jobs 3, 4, and 7 will be regarded as the neighboring jobs in order 2. Job 7 will be regarded as the only job in order 3, which has no other neighboring job.

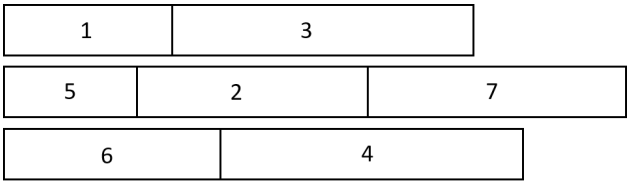

Figure 2: A description of the neighborhood jobs

In each iteration, we define the scope of neighboring jobs and swap them to see if the schedule after the exchange will have lower penalty or not. The scope of neighboring job is the order number we consider to switch. For example, if the scope of neighboring job is three, we will consider to exchange jobs with the three orders around it. Every solution after the swap which has been applied in the recent past will be recorded in a Tabu list. If the solution is already in the Tabu list, we do not need to swap them again. However, the solutions will only be recorded in the Tabu list for a limited time. Since the Tabu list has a maximum size which called Tabu size, when the numbers of swaps exceed the Tabu size, the oldest solution will be forgotten. The algorithm stops when it finished a certain number of consecutive iterations.

5 Numerical study

5.1 Experiment setting

We conduct extensive numerical studies to observe the performance of our proposed algorithm. For each machine, we set the production rate A_i and the lower bound of yield rate L_i as a fixed number and set the yield declining rate B_i and the initial yield rate $r_{i,0}$ as a number which is randomly distributed in a certain interval. Each job has a certain demand quantity Q_j and due time D_j, which are also randomly distributed in a certain interval. To simplify the problem, we set the maximum number of machines under maintenance in the same period H as many as n_M. In other words, we do not need to consider the limited resource of maintenance such as human or material resources. We also set the maintenance

time of each machine F_i to 1, so each maintenance will only cost one period of time.

To present the experimental results of the algorithm, we adopt two factors to analyze the performance under different circumstances. The first factor is the number of machines, which we set $n_M = 2$, $n_M = 3$, and $n_M = 5$ as three different levels. The second factor is the number of jobs in the scheduling plan, and we consider three levels: $n_N = 15$, $n_N = 25$,and $n_N = 50$.

We then propose six versions of algorithm, each with a different value of the Tabu size or a different job listing procedure. For Tabu size, we try 50 and 100. The above factors generate $3 \times 2 = 6$ scenarios, and we generate 20 instances for the study. All instances are computed with Gurobi solver and our algorithm. The experiments were performed on a desktop with a 3.6 GHz Intel(R) Core i9-9900K processor and 16 GB RAM. The heuristic algorithm was implemented using python 3.8. And the MIP model was solved using Gurobi Optimizer 9.1 and implemented with the Python programming language.

To compute the performance of our algorithm, we run the Gurobi model for comparison. Since the MIP model costs too much time to solve our problem, we set 1800 seconds as the time limit for each instance. The theoretical lower bound returned by Gurobi Optimizer when the time limit is reached is our first candidate of the lower bound. We also construct a relaxed mathematical model to solve the problem in reasonable time. We relax the binary variable y_{ijt} to be fractional, which means each machine can process multiple jobs in one time period. For most instances, 1800 seconds are enough for the relaxed model to generate an optimal solution to it. This serves as our second candidate of the lower bound. The final lower bound we use for a given instance is then the larger one of the two candidates. This lower bound is then used to be compared with the objective value of the solution generated by our proposed algorithm.

5.2 Solution performance

In this section, we demonstrate the best result in our experiments. According to the experiment result, we find that the last listing approach , which considers the available time of the

machine according to the ratio defined in (20), has the best performance. Regarding the parameters we test in the algorithm, setting the Tabu size to 100 and the scope of neighboring jobs to 5 results in the best performance. We use z^{LIST} to denote the objective value of the initial solution obtained after the job listing and maintenance insertion procedure, z^{TABU} to denote that of the final solution reported by our algorithm, and z^* to denote the theoretical lower bound we defined previously.

In Table 3, we observe that the average performance gap is 15.43% between the Tabu algorithm solution and the solver solution. The performance of our algorithm becomes worse as long as the number of jobs increases. This is because the more jobs we need to dispatch, the more complex problem is. The complexity of the problem will affect the number of possible solutions and make the optimal solution more difficult to find. The performance also increases along with the machine number because the shortage decreases if the machine number increases. On the other hand, we find that the processing time of our algorithm is always much lower than that of the MIP model. The average time for our algorithm to solve an instance is 166.8303 seconds, which is much faster than 1755.5898 seconds by the solver.

6 Conclusion

In this study, we consider a job scheduling problem as maintenance. We formulate the problem into a mixed integer problem as one way to generate a schedule. Since the problem is NP-hard, we develop a heuristic algorithm. We conclude that our proposed algorithm may find a near-optimal solution within a reasonable amount of time.

Though our model and algorithm consider the number of machines and jobs as factors, there are still other factors that may be considered. For example, the decision maker may consider different distributions of the due time, which represent the urgency of jobs. The production rate and yield decay rate also affect the influence of maintenance. To get closer to practice, we may use a more realistic function to calculate the yield rate in the future.

Scenario		Optimality gap		Total processing time (sec)		
m	n	$\frac{z^{LIST}}{z^*} - 1$	$\frac{z^{TABU}}{z^*} - 1$	LIST	TABU	MIP
2	15	28.11%	10.09%	0.0014	5.8167	>1752.5352
2	25	44.31%	15.37%	0.0025	34.5147	>1800.0536
2	50	69.72%	39.01%	0.0059	348.5633	>1800.0840
3	15	17.80%	7.54%	0.0018	6.3726	>1751.6196
3	25	37.38%	12.64%	0.0025	42.5319	>1800.0497
3	50	60.18%	24.03%	0.0048	462.1541	>1800.0696
5	15	13.75%	7.07%	0.0016	5.3166	>1495.7796
5	25	22.44%	10.21%	0.0029	46.5009	>1800.0585
5	50	42.94%	12.94%	0.0054	549.7016	>1800.0586
average		37.40%	15.43%	0.0032	166.8303	>1755.5898

Table 3: Numerical result of all scenarios.

References

Aghezzaf, E.H., M.A. Jamali, D. Ait-Kadi. 2007. An integrated production and preventive maintenance planning model. *European Journal of Operational Research* **181**(2) 679–685.

Aghezzaf, E.H., N.M. Najid. 2008. Integrated production planning and preventive maintenance in deteriorating production systems. *Information Sciences* **178**(17) 3382–3392.

Cassady, C.R., E. Kutanoglu. 2005. Integrating preventive maintenance planning and production scheduling for a single machine. *IEEE Transactions on Reliability* **54**(2) 304–309.

Kubzin, M.A., V.A. Strusevich. 2006. Planning machine maintenance in two-machine shop scheduling. *Operations Research* **54**(4) 789–800.

Lee, C.Y. 1996. Machine scheduling with an availability constraint. *Journal of Global Optimization* **9**(3-4) 395–416.

Lee, C.Y., Z.L. Chen. 2000. Scheduling jobs and maintenance activities on parallel machines. *Naval Research Logistics (NRL)* **47**(2) 145–165.

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

Mobley, R.K. 2002. *An introduction to predictive maintenance*. Elsevier.

Najid, N.M., M. Alaoui-Selsouli, A. Mohafid. 2011. An integrated production and maintenance planning model with time windows and shortage cost. *International Journal of Production Research* **49**(8) 2265–2283.

Rezg, N., X. Xie, Y. Mati. 2004. Joint optimization of preventive maintenance and inventory control in a production line using simulation. *International Journal of Production Research* **42**(10) 2029–2046.

Schmidt, G. 2000. Scheduling with limited machine availability. *European Journal of Operational Research* **121**(1) 1–15.

Ullman, J.D. 1975. Np-complete scheduling problems. *Journal of Computer and System sciences* **10**(3) 384–393.

Wang, H. 2002. A survey of maintenance policies of deteriorating systems. *European Journal of Operational Research* **139**(3) 469–489.