Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844
Vol. IV (2009), No. 4, pp. 439-451

MPM Job-shop under Availability Constraints

N. Zribi, L. Duţă, A. El Kamel

Nozha Zribi
École Centrale de Lille
France

Luminiţa Duţă
State University "VALAHIA Tirgoviste", Romania

Abstract: A large part of scheduling literature assumes that machines are available all the time. In this paper, the MPM Job-shop scheduling problem, where the machine maintenance has to be performed within certain time intervals inducing machine unavailability, is studied. Two approaches to solve the problem are proposed. The first is a two-phase approach where the assignment and the sequencing are solved separately. The second is an integrated approach based on the exact resolution of the 2-job problem using the geometric approach.
Keywords: genetic algorithm, geometric approach, assignment heuristic

Most of scheduling literature assumes that machines are available all the time. However, in many realistic situations, e.g., in typical industrial settings, machine breakdowns and preventive scheduled maintenance have rather quietly common occurrences. These considerations increase the complexity of any scheduling problem but make the problem closer to the industrial reality [26].

In this paper, we consider the job-shop scheduling with Multi Purpose Machines and availability constraints. We consider the deterministic model where the unavailability periods corresponding to maintenance tasks are known in advance. We also assume that preemption of operations is not allowed. More precisely, an operation O_{ij} of job J_i on machine M_k starts only if its execution can be finished before M_k becomes unavailable. The problem considered is a generalization of the classical job-shop problem and the multi-purpose machine problem studied in [12], where machines are available all times.

As compared to the literature dedicated to classical scheduling problems, studies dealing with limited machine scheduling problems are rather rare. Availability constraints have been firstly introduced in single machine [1], [28] and parallel machines [24], [25]. Lee extensively investigated flow-shop scheduling problems with two machines [15], [18], [19]. In particular, the author defined the resumable, non-resumable and semi-resumable models. An operation is called resumable if it can be interrupted by an unavailability period and completed without penalty as soon as the machine becomes available again. If the part of the operation that has been processed before the unavailability period must be partially (respectively fully) re-executed, then the operation is called semi-resumable (respectively non-resumable). Recently, flow-shop scheduling problems with two machines and resumable jobs have been treated in [9] and [14]. Job-shop problem under unavailability constraints has also been considered recently [30], [3] where authors proposed a branch and bound algorithm for the job-shop problem with heads and tails and unavailability periods. The problem considered here is strongly NP-hard since problem without unavailability periods is already strongly NP-hard [12]. In this paper we propose two different approaches to solve this problem.

The remainder of this paper is organized as follows. After a description of the considered problem in the following section, we propose first a two-phase method where a heuristic is used to solve the assignment problem and a genetic algorithm is developed for the sequencing problem. An integrated method, based on the exact resolution of the 2-job problem, is then developed. A comparison between the two algorithms is given in section 4.

1 Problem formulation

The MPM job-shop (job shop with Multi Purpose Machines) with availability constraints ($J(MPM)NCwin \mid C_{max}$: terminology defined in [29]) may be formulated as follows. There are n jobs $J_1,...,J_n$ to be processed on a set of m machines $R = (M_1,...,M_m)$. Each machine M_r can process at most one job at a time. Each job J_i consists of a sequence of n_i operations, that must be accomplished according to its manufacturing process. Each operation O_{ij} $(i = 1,...,n; j = 1,...,n_i)$ can be performed by any machine M_r in a given set $\mu_{ij} \subset R$ for p_{ij} time units. Each operation is non-preemptive, i.e., it must be accomplished without interruption. Moreover, we assume that machine M_r is unavailable during giving periods corresponding to preventive maintenance. The starting times and durations of these tasks are fixed and known in advance. We note K_r the number of maintenance tasks on machine M_r. A_{rl} and D_{rl} represent respectively the starting and the finishing time of the l^{th} maintenance task on machine M_r.
The objective is to find a schedule, defined by the starting time S_{ij} and the completion time C_{ij} of each operation O_{ij}, with a minimum makespan ($maxC_{ij}$).

The scheduling problem in $J(MPM)NCwin \mid C_{max}$ can be decomposed in two subproblems:

- a routing subproblem that consists in assigning operations to machines;
- an operation scheduling subproblem associated with each machine to minimize the makespan. This is a Job-Shop scheduling Problem with Availability Constraints $J,NCwin \mid Cmax$.

2 Two-phase approach for the problem $J(MPM)NCwin \mid C_{max}$

2.1 The routing problem

Since the precedence constraints could be relaxed following the decomposition of the problem in two separate stages, the assignment problem may be treated as a parallel machine problem with the two additional constraints:

- an operation can be performed by a machine belonging to a subset of the set of the available machines: partial flexibility
- machines are subjected to several maintenance constraints: the planning horizon is divided into subintervals

We propose a heuristic based on several priority rules taking into account these two additional constraints.

Assignment heuristic

We use a list algorithm based on priority rules in order to construct an initial assignment solution.
Let us define the following parameters:

- r_{ij}: earliest starting time of operation O_{ij} (definition 1)
- T_k^s availability date of machine M_k at iteration s, where s denotes the iteration number
- ER_k set of operations which can be performed on machine M_k
- EA_k^s set of operations which can be assigned to machine M_k at iteration s
- CM_k^s load of machine M_k at iteration s

Definition 1. To each operation O_{ij}, we associate an earliest starting time r_{ij} calculated by the following formula:

$$\begin{cases} r_{i,1} = 0 \ \forall \ 1 \leq i \leq n \ , r_{i,j+1} = r_{i,j} + p_{i,j} \\ \quad \forall \ 1 \leq j \leq n_i - 1, \ \forall \ 1 \leq i \leq n. \end{cases} \quad (1)$$

- In step 1, the different parameters are initialized.
- In step 2, for each machine we determine the set EA_k^s of operations such that: $r_{ij} \leq T_k^s$.
- In step 3, we evaluate (the potential) starting time of each operation on each possible machine. The availability periods are taken into account. In fact we test if the operation can be scheduled before the next availability period on that machine.

A pair (operation/machine) is selected using the following priority rule:

The less flexible machine M_k ($Min\ CM_k^s$) is selected. Operations in EA_k are sorted in non decreasing order of card(μ_{ij}) (priority is accorded to the less flexible operation). M_k is assigned to the first operation in EA_k which can be scheduled before the next unavailability period.

This priority rule allows occupying the time intervals before the unavailability periods and takes into account the load of machines and the flexibility degree of each operation.

In order to ensure a high level of the solution quality, we have chosen to improve the assignment given by the assignment heuristic. To this end, a local improvement search has been studied. Such search is based on a Tabu algorithm , an adapted routing move technique and an adapted criteria for the studied problem. In next section, we give a description of the Tabu algorithm.

A Tabu Search Algorithm for the assignment problem

Optimization criteria: For a classical routing problem, where machines are available all times, we choose, in general, to minimize the workload of the most loaded machine, since it provides a lower bound for the makespan.

We define, for each assignment S, a lower bound denoted $LB(S)$ for the makespan corresponding to S. This lower bound is based on relaxation into a set of single-machine problems taking into account the unavailability periods. The objective of the tabu search algorithm presented here is to minimize $Cr_1 = LB(S)$ and hence to preoptimize the makespan.

Given an assignment S, we associate with each machine M_k, a single-machine problem π_k with ready times (definition 1), tails (definition 2) and unavailability periods.

A lower bound for π_k is the makespan of a preemptive schedule with unavailability periods, based on the Jackson Preemptive Schedule (JPS) algorithm. Such schedule is calculated for each machine and $LB(S)$ is the maximum makespan value of these schedules.

Definition 2. After the finishing of operation O_{ij} , a time of q_{ij} has to go before job J_i is finished completely. q_{ij} is called the tail of operation O_{ij}

The procedure of preemptive schedule allows constructing the optimal schedule when preempt-resume applies and hence to obtain a lower bound for π_k due to the two following reasons:

1. The unavailability period is treated as an operation, so the problem here is equivalent to the preempt-resume case where JPS gives the optimal solution.
2. The unavailability period will start right on its ready time and will never be preempted since it has the largest tail among the available operations. Preemptive schedule is calculated for each machine and $LB(S)$ is the maximum makespan value of these schedules.

Description of the Tabu Search (TS) algorithm TS was introduced by Glover as a general iterative meta-heuristic for solving combinatorial optimization problems [16]. The TS algorithm is as follows.

- The initial solution is obtained by applying the assignment heuristic described above.
- The solution is described as a list of operations with their corresponding machines.
- A routing move is defined by the relocation of a critical operation (operation that belongs to the critical machine) to a feasible machine position. For a given solution, we consider every possible relocation of every reroutable critical operation.
  The routing move is based on the following steps:
  1. Find the critical machine M_{k_c}.
  2. Find an operation O_{ij} that can be assigned to another machine $M_{k_o} \in \mu_{ij}$ without increasing the criterion value.
  3. Reassign O_{ij} to M_{k_o} if possible.
- The Tabu list consists of pairs $(op;m_o)$, where op denotes the operation that is moved from machine m_o to a different machine.
- The choice of the move is based on the value of Cr_1 which is the maximum makespan value of the preemptive schedules.

2.2 Genetic Algorithm for the sequencing problem

After the assignment step, each operation is assigned to a fixed machine. Thus the MPM job-shop problem is reduced to a job-shop problem with availability constraints (JSPAC).

The problem is then to assign a starting time S_{ij} and a completion time C_{ij} to each operation O_{ij} ($C_{ij} = S_{ij} + p_{ij}$). The considered objective is to minimize the makespan ($C_{max} = max_{i,j} C_{ij}$). We propose a Genetic algorithm to optimize the makespan in a JSPAC.

Coding

According to the literature, two types of approaches exist. In the first, the schedule is directly coded in the chromosome. In the second, a scheduler is associated to the GA to transform the chromosome into actual schedule. In this paper, the latter approach is used to code GA chromosomes. In fact, we use a representation based on job operation. It consists in representing the schedule in a chain of NT operations ($\sum_{1 \leq i \leq n} n_i$) where operations of the same job are represented by the same symbol J_i, the job number. Each job J_i appears exactly n_i times (n_i is the number of operations of J_i) in the chain. For example , for a job-shop problem of dimension 3*3 (3 jobs and 3 operations per job), an example of a chromosome is (1 2 1 3 1 3 2 2 3).

The computation of the starting time and the completion time (S_{ij} , C_{ij}) is obtained according to the order z of each task in the chain (chromosome) and taking into account the unavailability periods of the machines.

Crossover and mutation

We use GOX - Generalized Order Crossover , a swap based mutation and an "Intelligent" mutation operator.

"Intelligent" mutation operator: This operator consists in reducing the idle time before unavailability periods in order to improve the makespan.

This mutation heuristic is described by algorithm 1 and consists in:

For each unavailability period of the critical machine, we exchange an operation scheduled before a maintenance period with another operation which can begin before this maintenance period, but is scheduled after this one. All possible permutations are tested. The permutation minimizing the makespan is selected.

Algorithm 1 Heuristic for the "intelligent" mutation operator

Choose a chromosome X aleatory
calculate the schedule and find the critical machine M_{max}
find OM_{max} : set of operations O_{ij} assigned to M_{max}
For each maintenance period $\mathcal{D}$ of M_{max}
  For each operation O_{ij} in OM_{max} finishing before $\mathcal{D}$
    For each operation $O_{i'j'}$ de OM_{max} which can begin before $\mathcal{D}$ and having been scheduled after $\mathcal{D}$
      Test the permutation of O_{ij} and $O_{i'j'}$
Choose the permutation minimizing the makespan

2.3 Application example

Let us consider an example of a MPM job-shop. It is made of 15 jobs, 5 operations per job and 5 machines. Each machine is subject to two maintenance periods as follows:

M_1: [201 250],[463 512] means that the machine M_1 is unavailable between the dates 201 and 250 and between the date 463 et 512.
M_2: [104 139] , [520 588]
M_3: [233 331], [499 528]
M_4: [137 186], [507 556]
M_5: [129 187], [783 881]

1. Assignment step

   We apply first the assignment heuristic to obtain an initial solution.

   In order to evaluate the solution given by the assignment heuristic, we report in table 1 the load of the different machines as well as the makespan value of the preemptive schedule constructed using Jackson rule.

	M_1	M_2	M_3	M_4	M_5
load	714	782	868	789	839
preemptive schedule	812	907	995	887	997

Table 1: Results of the assignment schedule

2. Sequencing problem

	M_1	M_2	M_3	M_4	M_5
load	815	812	790	818	757
preemptive schedule	913	915	917	916	915

Table 2: Results of the TS algorithm

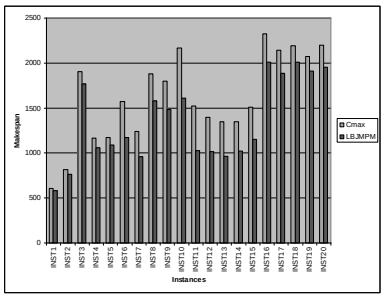

Figure 1: Simulation tests

Table 3 gives the value of the lower bounds based on preemptive schedule for the initial solution of assignment (solution 1) and the solution obtained after applying TS (solution 2) as well as the makespan found by the genetic algorithm.
These results show that TS improves the global solution of the problem and GA gives a good solution comparing to the lower bound. We propose in the next section an integrated approach based on the exact resolution of the 2-job problem.

	solution 1	solution 2
preemptive schedule	997	917
Makespan	1023	968

Table 3: Results of the genetic algorithm

Other experiments were performed on randomly generated instances with more or less availability periods. We report in this section different results for three classes of instances. Each class has five instances. The number of jobs for each class is 10 , 15, 10 and 20 respectively. The number of machines is equal to 5 , 10, 10 and 10 respectively. The processing time for each operation is randomly selected in [|40,150|]. For each machine, the maintenance tasks occur after at least one operation. The starting time for each maintenance task differs from machine to another. The duration of a maintenance task on a machine is the average of the processing times of operations. For the GA, the mutation and crossover probabilities are fixed to: ($P_{crossover} = 0.8$, $P_{mutation} = 0.2$).

In figure 1 we compare the result of the two-phases approach with a lower bound denoted LB_{JMPM} that we have developed for the problem [31].

Comparing with LB_{JMPMAC} our approach gives interesting results (RD_{JMPMAC} 20.6%) with a short computational time. It's worth noting that LB_{JMPM}, which is based on a lower bound for the parallel machine problem, is more interesting for instances with high flexibility.

3 An integrated approach for the $J(MPM),NCwin \mid C_{max}$

This approach is based on the exact resolution of the problem $J(MPM),N_{Cwin}|n=2|Cmax$. A polynomial algorithm is developed to solve the problem.

3.1 Polynomial algorithm for $J,N_{Cwin}|n=2|Cmax$

State of the art

The geometric approach has been firstly introduced by Akers and Friedman (1955) in [2]. It consists in reducing the two-job job-shop scheduling problem in the search of a shortest path and thus gives a polynomial algorithm to solve it.

The first step of the geometric approach is the representation of the scheduling problem in a 2-dimensional plane with obstacles, which represent the machine conflict between operations of the two jobs [22]. More precisely,

- Each job J_i is represented by an axe with n_i intervals according to its manufacturing process.
- Each interval corresponds to an operation O_{ij} and has a length of p_{ij} (fig.2).
- Intervals O_{1j} and O_{2k} form an obstacle if O_{1j} and O_{2k} share the same machine (fig.2).
- The horizontal and the vertical crossing the final point F, which corresponds to the completion of the two jobs, are considered as the final obstacle.

A feasible solution of the scheduling problem is then a path going from the origin O to the final point F. Such a path consists of horizontal, vertical and diagonal legs. A horizontal (resp. vertical) leg represents the exclusive progression of job J_1 (resp. J_2), whereas diagonal legs correspond to simultaneous executions of the two jobs. Moreover, any path must avoid the interior of the obstacles. This is due to the fact that two operations can not be executed simultaneously on the same machine and are not preemptable. The length of a horizontal or vertical segment is equal to its usual length while the length of a diagonal segment is equal to the length of its projection in any axe, which is the time spent for the simultaneous processing of two operations.

The shortest path problem in the plane can be transformed into an unrestricted shortest path problem in an acyclic network (see fig.2), where the set of vertices corresponds to the origin O, the final point F and the North-West and South-East corners of the obstacles. Each vertex has at most two successors obtained by going diagonally until hitting an obstacle D. If the obstacle D is the final obstacle, the vertex F is the only successor of node, otherwise the NW and SE corners of obstacle D, are immediate successors of node (see fig.2).

The Temporized Geometric Approach (TGA), developed by Aggoune [3], [4] is an extension of the geometric approach which exactly solve the problem $J,N_{Cwin}|n=2|Cmax$. It allows integrating the evolution of time and so the availability of the machines, based on the definition and the introduction of new vertices, as well as a new and dynamic way to progress from one vertex to its successors.

Vertices Characterization and Definitions in TGA: In the classical geometric approach, vertices of the network are the north-west (NW) and south-east (SE) corners of the obstacles hit when going diagonally in the plane. These corners are located at the extremities of the intervals corresponding to operations in conflict. Each vertex can then be defined thanks to its coordinates in the plane: the x-coordinate (resp.

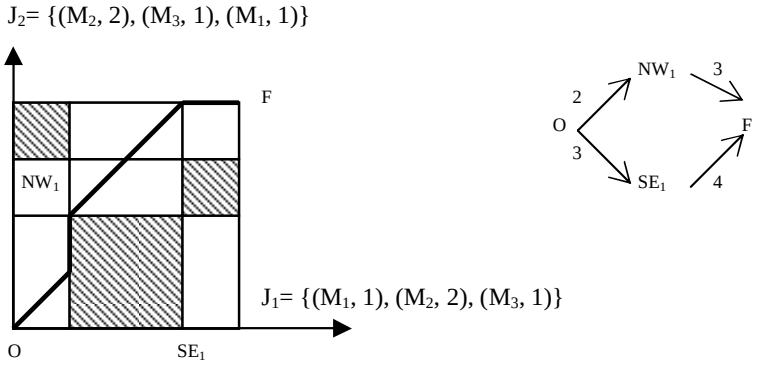

Figure 2: Classic geometric approach

y-coordinate) of the vertex corresponds to operation of job J_1 (resp. J_2) to be executed. In TGA [4], some vertices can be located between two lines bounding an operation, i.e., inside the intervals. For each vertex, each coordinate is additionally attributed by information related to the duration of already processed part of the associated operation. Moreover, an earliest starting time $h(S)$ is associated with each vertex S. $h(S)$is the length of the shortest path from the origin to S.

The set of vertices of the network constructed by TGA is composed by the three following types of vertices:

- Regular vertices, located at the intersection of horizontal and vertical lines, to which *NW* and *SE* corners of obstacles belong.
- Singular vertices, located on a horizontal (resp. vertical) line, which means that the execution of the operation of job J_2 (resp. J_1) has not started yet
- Waiting vertices, also located at the intersection of two lines, for which the execution of operations of jobs J_1 and J_2 has not started yet.

A singular vertex is created if the progression of only one job is possible (availability problem for the other job), whereas waiting vertices are created if the progression is possible for none of the two jobs. A waiting vertex is always a duplication of the regular vertex having the same geometric coordinates but not the same earliest starting time. The progression works as follow:

- If the operations of the two jobs cannot start at time $h(S)$, the earliest starting time of vertex S, a waiting vertex is then created.
- If there is an availability problem in the direction J_1 (resp. J_2), the progression is made along the vertical (resp. horizontal) line, what means that the execution of the operations of job J_2 (resp. J_1) only, until job J_1 (resp. J_2) becomes available. A singular vertex , from which a diagonal progression is possible, is added as successor of S.
- If there is no availability problem, that is to say if the operations of the two jobs can be executed at time $h(S)$, the progression works as in the classical geometric approach.

3.2 An extended approach

We propose a generalization of TGA (GTGA) in order to deal with the flexibility property of the $J(MPM), NCwin \mid n = 2 \mid C_{max}$. This generalization is based on the works of Aggoune [4] and Mati and al. [21] for the flexible Job-shop without availability constraints. As for the job-shop problem, the

scheduling of MPM job shop can be represented in the 2-dimensional plane with potential obstacles that depend on the assignment of machines for the two jobs. Let us define the vertices of the network describing the progression in the plane, the successors of each vertex and the distance between any two vertices. We develop the algorithm *SuccVertex* allowing to find the successors of each vertex $S = ((k_1, \Delta^1_{k_1}), (k_2, \Delta^2_{k_2}))$. The algorithm *SuccVertex* is in three steps.

- Step 1 is an initialization step. Set P_h (resp P_v) is defined to keep the machines of job J_2 (resp J_1) allowing to progress until meeting an horizontal and $\backslash$ or a vertical. This set is used to progress in the next iteration of the program in the case of diagonal progression. E_1 is the set of the possible machines of O_{1v} and E_2 is the set of the possible machines of O_{2h}.

- In step 2, first we check the availability of the two machines for O_{1v} et O_{2h}. If these machines are available, we progress diagonally until a vertical (end of operation of job 1) and or an horizontal (end of operation of job 2) is met depending on the duration of the current operations. Set P_h or P_v is defined to keep the machines of job J_2 or J_1 respectively in the case of diagonal progression. If an availability problem occurs in one of the two directions, the algorithm *SuccVertexAvailability* is used to define the successors of S in this case. The machines concerned by this case are memorized in sets $P_{v\_}\ availavility$, $R_{h\_}\ availability$, $P_{h\_}\ availavility$ and $R_{v\_}\ availability$. These sets are used by algorithm *SuccVertexAvailability*. The corners *SE* and *NW* of the potential obstacles that could be reached from S are added as successors of S during the diagonal progression.

- In step 3, we update the current time $current\_\ time$, the sets E_1, E_2, v and h.

The algorithm *SuccVertex* is stopped if the final obstacle is hit or when the diagonal progression is not possible because of availability problems or unavoidable obstacle. The algorithm *SuccVertexAvailability* allows progressing horizontally or vertically (availability problem) until the progression in the two directions becomes possible (end of the unavailability period) , in this case a singular or a regular vertex is added as successor to S ; another availability problem occurs, and in this case a waiting vertex is added as successor of S or an unavoidable obstacle is hit.

Distance Between Two Vertices S and S':

The distance between a vertex v_i and its successor $v_{i'}$ is calculated inside the two developed algorithms using the variable $current\_\ time$. In fact, all machines used to progress from v_i until $v_{i'}$ are fixed. Remark: If F is a successor of S, the distance between S and F is calculated using only available machines, if possible, or machines becoming available first. In fact we neglect the other paths.

Theorem 1. The set of vertices constructed by applying algorithm SuccVertex is sufficient to determine the optimal schedule.

Proof. The correctness of theorem 1 is due to the fact that TGA gives the optimal schedule in the case of classical job-shop [3] and the developed algorithm checks all possible machines for each operation.

3.3 The general job shop problem with multi purpose machine and availability constraints

From the result of the previous section we can deduce a greedy heuristic to calculate a solution for $J(MPM), NCwin \mid Cmax$. This heuristic works as follows:

1. The two first jobs are optimally scheduled using the GTGA algorithm.

2. Additional unavailability periods, corresponding to the execution of operations of the two scheduled jobs, are fixed on each machine.

	Instance size (*nxmxNT*)	Two-phase approach	Integrated approach
Inst1	10x5x50	852	833
Inst2	10x5x50	641	631
Inst3	10x5x50	697	707
Inst4	10x5x50	710	703
Inst5	10x5x50	633	625
Inst6	15x5x75	945	983
Inst7	15x5x75	897	891
Inst8	15x5x75	1023	1017
Inst9	15x5x75	1001	997
Inst10	15x5x75	998	1002

Table 4: Comparison of the two approaches

3. The algorithm is applied to the next two jobs of the sequence, taking into account the initial and the new unavailability periods. This procedure continues until all jobs are treated.

If the number of job is odd we need an insertion procedure to schedule the last job. It is based on the following rule:
An operation of the last job is inserted in such a way, that it begins as early as possible, if we have the choice between two machines we choose the machine giving the smallest idle time.

4 Experimental results

To perform an experimental evaluation of the proposed approaches, we present, in this paper, a ten classical flexible job-shop instances [13]. In order to provide proper experimental settings, two availability periods are generated randomly for each machine. It is worth noting that these results were also confirmed by several other experiments based on randomly generated instances with more or less sizes and/or availability periods [31].

In table 4, we report the values of the makespan given respectively by the two-phase approach and the integrated approach.

These simulations show that the two-phase approach gives interesting results comparing with the integrated approach. The main advantage of the two-phase approach is related to the computation time.

The integrated approach is rather more complicated mainly in terms of computing time. Besides, we note that solutions given by the integrated approach vary with the initial sequence of jobs. Hence, this approach can be improved by adding an optimization algorithm for the initial sequence of jobs. Meanwhile, it appears clearly worth applying this polynomial algorithm in a Branch and Bound algorithm, as done by Jurisch for the classical MPM job-shop without availability constraints [17].

5 Conclusion

We have investigated in this paper MPM job shop scheduling problems under availability constraints. We have proposed two kinds of methods. The first one solves the assignment and the sequencing problems separately. The second one is based on an extension of the geometric approach to deal with the machine availability and the flexibility property of the problem. We are now working on the development of a maintenance management system with an industrial partner including these optimization in a multi-objective environnement. In fact, The analysis of performance of scheduling problem involves

more than one criteria [7]. We are focusing on two main criteria : the maintenance total costs and the delivery processes. Other aspects that we will focus on is the real-time approaches to solve schedule problem with unanticipated interruptions [5].

Bibliography

[1] Adiri I., Bruno J., Frostig E., and Rinnooy Kan A. H. G. : Single machine flow-time with a single breakdown. Acta Informatica, 26, 679-696.

[2] S.B. Akers et J. Friedman. – A non-numerical approach to production scheduling problems. *Operations Research*, vol. 3, 1955, pp. 3, 429–442.

[3] Aggoune, R.: Ordonnancement d'Ateliers sous Contraintes de Disponibilité des Machines(2002) Ph.D. Thesis, Université de Metz, France.

[4] Aggoune, R.: Two-job shop Scheduling Problems with availability Constraints. ICAPS 2004, June 3-7, 2004, Whistler, British Columbia, Canada.

[5] Duta, L.: Contribution ŕ l'étude de la conduite des systčmes de désassemblage.These (2006). Université de Franche-Comté Université de Bucarest.

[6] Duta,L., Filip, F. G. Henrioud,J.-M. Popescu: Disassembly Line Scheduling with Genetic Algorithms. Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844 Vol. III (2008), No. 3, pp. 270-280.

[7] Filip F.G., Neagu G. and Donciulescu D. (1983). Job shop scheduling optimization in real time production control. Computers in Industry, 4(4) (North Holland, Amsterdam), 395 - 403.

[8] Ruiz-Torres AJ. , Nakatani K.: Application of real-time simulation to assign due dates on logistic-manufacturing networks. Proceedings of the 1998 Winter Simulation Conference.

[9] Blazewicz J., Breit J., Formanowicz P., Kubiak W., Schmidt G.: Heuristic algorithms for the two-machine flowshop problem with limited machine availability. Omega Journal (2001), 29, 599-608.

[10] Carlier J.: Scheduling jobs with release dates and tails on identical machines to minimize the makespan. European Journal of Operational research(1987), 29:298-306, North-Holland.

[11] Carlier J.: The One-Machine Sequencing Problem, European

Jounal of operational research(1982), 11: 42-47.

[12] Jurisch B.: Scheduling Jobs in Shops with Multi Purpose Machines (1992) Ph.D Thesis, Universität Osnabrück.

[13] J. Hurink, B. Jurisch et M. Thole. – Tabu search for the job-shop scheduling problem with multi-purpose machines. *Operations Research Spektrum*, vol. 15, 1994, pp. 205–215.

[14] Kubiak W., Blazewicz J., Formanowicz P., Breit J., Schmidt G.: Two-machine flow shops with lim-ited machine availability. European Journal of Opera-tional Research (2002) 136: 528-540.

[15] Lee C.Y.: Machine scheduling with an availability constraint. Journal of Global Optimization(1996) 9: 395-416.

[16] Glover F., Laguna M.: Tabu search, Kluwer Publish- ers, Boston 1997.

[17] Jurisch, B., (1995). Lower bounds for the job-shop scheduling problem on multi-purpose machines. Discrete Applied Mathematics, 58, 145-156.

[18] Lee C.Y.: Minimizing the mazkespan in two-machine flows-hop scheduling with availability constraint. Operations research letters (1997), 20: 129-139, .

[19] Lee C.Y.: Two-machine flowshop scheduling with availability constraints. European Journal of Opera-tional Research (1999), 114: 420-429.

[20] Lee KM., Yamakawa T.: A Genetic algorithm for general machine scheduling problems. Int.Conf. on Conventional and knowledge-Based Electronics Systems, Vol 2 pp60-66 Australia, 1998.

[21] Mati, Y., and Xie, X.: Un algorithme polonimial pour le job shop flexible avec blocage : cas de deux jobs. Mosim'2003, April 23-25 april, Toulouse, France.

[22] Mati, Y., and Xie, X.: The complexity of two-job shop problems with multi-purpose unrelated machines. European Journal of Operational Research, 152 (1), 159-169, 2004.

[23] Nowicki E., Smutnicki C.: A Fast Taboo Search Algorithm for the Job-Shop Problem, Management Science (1996), Vol. 42, No. 6, pp. 797-813.

[24] Schmidt G.: Scheduling on semi identical processors. Z. Oper.Res.1984, A28, 153-162, .

[25] Schmidt G.: Scheduling independent tasks with deadlines on semi-identical processors. Journal of Operational research society, 39, 271-277, 1988.

[26] Sanjay J. , William J. Foley :Impact of Interruptions of Schedule Execution in Flexible Manufacturing Systems. The International Journal of Flexible Manufacturing Systems, 14, 319-344, 2002.

[27] Roy B., Sussmann B.: Les probl'emes d'ordonnancement avec contraintes disjonctives (in French). Technical Report 9 bis, SEMA, Paris (France), December 1964.

[28] Leon V. J., Wu S. D.: On scheduling with ready-times, due-dates and vacations. Naval Research Logistics, 39:53-65, 1992.

[29] Schmidt G.: Scheduling with limited machine availability. European Journal of Operational Research(2000), 121, 1-15.

[30] Mauguiere Ph., Billaut J-C., Bouquard J-L.: New single machine and job-shop scheduling problems with availability constraints. Journal of Scheduling, 2004.

[31] Zribi N. : Ordonnancement des job-shop flexibles sous contraintes de disponibilité des machines. PHD Thesis, Ecole Centrale de Lille, France, 2005.

Nozha Zribi has obtained an engineer diploma in 2002, a Master of Science and European Master in Computer Engineering in 2002, and a PhD in operational research from Ecole Centale of Lille in 2005. Since 2007, she is a research engineer. Her interests are logistics, software design and implementation and SEA.

Luminiţa Duţă is currently associate professor (lecturer) at "Valahia" University of Targoviste, Romania. She took the PhD from Université Franche Comté, Besancon (France) and from Technical University "Politehnica" of Bucharest in Automation and Control field. Her research interests include the use of decision support systems in complex process controlling and meta-heuristics. She is a member of IFAC Technical Committees 5.2 and 5.4.

Abdelkader El Kamel (M'96-SM'00) received the engineering diploma in 1990, the Ph.D. in 1993 and the "Habilitation ŕ Diriger des Recherches" in 2000. He is currently Professor at the Ecole Centrale de Lille and Visiting Professor in different countries (China, Chili, India and Tunisia). He is the author or coauthor of more than 100 journal articles, book chapters, plenary sessions, and communications in international conferences. His current research interests include intelligent control of complex systems under uncertainty; modeling, design, real-time monitoring of autonomous dynamical systems and mobile cooperative robots. Dr. El Kamel was the President of the IEEE-SMC Conference held in Tunisia in 2002. He was Program chair or IPC member of several IEEE, IFAC and WAC conferences, Member of the AdCom of IEEE France section, President of the IEEE-SMC TC on "complex systems under uncertainty", Member of the IFAC TC 3.2 and President of the IEEE-SMC France Chapter.