World Academy of Science, Engineering and Technology
International Journal of Industrial and Manufacturing Engineering
Vol:9, No:10, 2015

A Genetic Algorithm to Schedule the Flow Shop Problem under Preventive Maintenance Activities

J. Kaabi, Y. Harrath

***Abstract*—**This paper studied the flow shop scheduling problem under machine availability constraints. The machines are subject to flexible preventive maintenance activities. The nonresumable scenario for the jobs was considered. That is, when a job is interrupted by an unavailability period of a machine it should be restarted from the beginning. The objective is to minimize the total tardiness time for the jobs and the advance/tardiness for the maintenance activities. To solve the problem, a genetic algorithm was developed and successfully tested and validated on many problem instances. The computational results showed that the new genetic algorithm outperforms another earlier proposed algorithm.

***Keywords*—**Flow shop scheduling, maintenance, genetic algorithm, priority rules.

I. Introduction

HAVING a consistent industrial system is one of the most important aims to guarantee a good quality of the manufactured products as well as the respect of the imposed deadlines. To attain such goals, the machines must be functional following an optimum working way according to a continuous survey and following periodic maintenance activities. Normally an efficient scheduling algorithm should take into consideration the availability of resources. However, most of the proposed algorithms in the literature assume that the machines are always available for processing jobs on time. Such a hypothesis makes the resulting schedule less realistic. In reality, when a machine breaks down or is under a periodic preventive maintenance, it stops performing its work and the cost doubles as a result of time waste that is, cost for lack of production and maintenance cost.

This paper studies the non-resumable flow shop scheduling problem under unavailability constraints. We suppose that the machines must be stopped to be regularly maintained according to predetermined intervals of time; i.e. preventive maintenance activities. In order to make our study more realistic, we allowed a maintenance activity to be delayed (or advanced) for a short period of time. However, an additional cost will be applied for any delay that increases the risk of machine breakdowns.

Most studies which treat the same type of problem consider only one criterion reflecting either a production cost or a time linked to the tardiness of jobs (in general the makespan Cmax). This is mainly due to the hypothesis considering that the maintenance activities are processed with neither delay nor advance. In our study, any delay or advance of a pre-arranged maintenance activity causes a right or left shift of the other tasks. That's why we included a new criterion that is an aggregation of the total earliness/tardiness time due to the maintenance activities and the total tardiness time of the jobs. Minimizing this new criterion helps processing the maintenance activities according to their predefined planning times. Few are the studies solving multi-objective scheduling problems under maintenance activities. A recent study achieved in [1] considered a multi-objective function optimizing Cmax as a criterion for the jobs and earliness/tardiness for the flexible maintenance activities in a job shop.

Few years ago, researchers started giving importance to the state of the machines when solving a scheduling problem. Usually, the machines are supposed to be in perfect states to perform their work at the needed time. The existing works which treat the availability constraints were published mainly for two-machine flow shop scheduling problems.

Reference [2] studied two-machine flow shop scheduling problem under machine availability constraints. The objective is to minimize the makespan. When the processing of a job is interrupted by an unavailability period of a machine, both the resumable scenario and the semi-resumable scenario were considered. For the problem with several unavailability intervals on the first machine under the resumable scenario, the authors proposed a fast (3/2)-approximation algorithm. However, for the problem with one unavailability interval under the semi-resumable scenario, a polynomial-time approximation scheme was developed. The same problem was relaxed in [3] by considering only one unavailability period on the first machine. The authors showed that the worst-case error bound ½ of the heuristic provided by [4] is tight. They developed an improved heuristic with a worst-case error bound of 1. Reference [5] investigated the same problem as in [4]. The authors developed two mixed-integer programming (MIP) models and proposed a branch and bound (B&B) algorithm based on a set of new lower bounds and heuristics. They showed that there is an impact of the unavailability start time period on the generated schedules.

A generalization of the problem, through the integration of several unavailability periods, was studied in [6]. The authors proposed two heuristics: one is based on Johnson's rule, and the other uses a local optimization. A simulated-annealing-based method was also developed.

Reference [7] published an optimal branch-and-bound-based method to solve the two-machine flow shop problem

J. Kaabi and is with the University of Bahrain, P.O. Box 32038 Kingdom of Bahrain (corresponding author phone: +973 36 53 09 56; fax: +973 17 44 91 19; e-mail: jkaapi@ uob.edu.bh).

Y. Harrath is with the University of Bahrain, P.O. Box 32038 Kingdom of Bahrain (e-mail: yharrath@ uob.edu.bh).

when both machines are subject to periodic break periods. For the particular case where only the first machine must be stopped, the authors proposed a heuristic with a complexity of O(n logn).

Reference [8] studied the two-stages flexible flow shop problem under availability constraints. The objective is to minimize the makespan. The authors showed that this problem is NP-hard even in the case where there is only one unavailability period. They considered one machine in the first stage with one unavailability period, and then proposed two algorithms with a small worst case bound.

Some other studies were achieved for the non-preemptive flow shop scheduling problem with m machines. Reference [9] studied two types of maintenance periods. The first type is when the starting times of the maintenance activities were fixed and known in advance. However, the second type is when the starting times of the maintenance activities were flexible. To solve this problem the author proposed a genetic algorithm and a hybrid Tabu search-based method.

Reference [10] investigated scheduling flexible flowshops subject to periodic preventive maintenance activities on machines. The objective is to minimize the makespan. The authors proposed two metaheuristics including a genetic algorithm and an artificial immune system. In addition, some constructive heuristics were developed. The results of the computational experiments indicated that the artificial immune system outperforms the other proposed methods. Although several methods, models and techniques have been developed to integrate production and maintenance activities, most of them considered only one criteria related to the production (Cmax). Few studies included a maintenance criterion. To the best of our knowledge, no study treated a criterion including both production tardiness and earliness/tardiness maintenance activities. This criterion will be presented in the next section.

II. Problem Formulation

The problem can be defined formally as follows: Given a set of *n* independent jobs $\{J_1, J_2, ..., J_n\}$ which are processed in a flow shop of *m* machines denoted by $M_1, M_2, ..., M_m$. Each job J_i has a release date r_i and a due date d_i. Each job J_i is assigned a processing time p_{ij} on machine M_j. The completion time of job J_i on machine M_j is denoted by C_{ij}. Each machine can perform only one job at a time and each job needs only one machine to be achieved. The preemption of the jobs is not allowed (whenever a machine starts a job it must finish it before processing the next one). The machines are subject to preventive maintenance activities. The periods during which the maintenance activities take place are predetermined in advance. However, the number of maintenance activities for each machine is not known in advance. The start times of maintenance activities are to be calculated during the scheduling procedure.

A. Maintenance Activity Notations

The notations used to describe a maintenance activity are defined as follows:

- D_j : maintenance activity processing time on machine M_j
- S_{kj} : start time of kth maintenance activity on machine M_j
- T_j : time interval between two consecutive maintenance activities on machine M_j
- ΔT_j : allowable tardiness/earliness time of any maintenance activity on machine M_j
- $T_{\min}^{j} = T_j - \Delta T_j$: minimum time between two successive maintenance activities on machine M_j
- N_j : number of maintenance activities on machine M_j
- w_{rj} : unit tardiness cost of a maintenance activity on machine M_j

B. Optimization Criterion

The main goal of our study is to schedule the maintenance activities with a minimum of tardiness time. The optimal schedule will assign these activities within their time intervals. The start time of any maintenance activity depends on its predecessors. Let

$$A_j = \sum_{k=1}^{N_j} \max(0, S_{(k-1)j} + D_j + T_{\min}^{j} - S_{kj}). \quad (1)$$

be the total earliness time, and

$$R_j = \sum_{k=1}^{N_j} \max(0, S_{kj} - T_{\max}^{j} - S_{(k-1)j} - D_j). \quad (2)$$

the total tardiness time of any maintenance activity on machine M_j.

In general, when customers order products they require deadline constraints to the suppliers. In other words, the products must be delivered on time. Consequently, it is better to consider the total tardiness time as an optimization criterion for the jobs.

The optimization criterion is an aggregation of both total earliness/tardiness time generated by the maintenance activities and the total tardiness caused by the jobs. The final optimization criterion to be minimized can be formulated as:

$$f = \sum_{i=1} \max(0, C_i - d_i) + \sum_{j=1} (A_j + w_{rj} R_j). \quad (3)$$

III. Solution Approach

We developed a genetic algorithm (GA_2) to solve the flow

shop scheduling problem where the machines are subject to preventive maintenance activities. We chose the genetic algorithm because of its ability to generate optimal or near to the optimal solutions for hard problems. This is mainly due to its global search aspect by the means of several genetic operators such as selection, crossover, and mutation. To develop a genetic algorithm, we should define the following elements:

- An encoding of solutions using a suitable alphabet to form chromosomes.
- A way to generate a non-homogenous first population.
- A fitness function to evaluate the different chromosomes based on the problem's objective function.
- A selection method of candidate chromosomes to be used for generating new chromosomes.
- Some genetic operators such as the crossover and mutation to produce new well adapted chromosomes and diversify the future populations. These operators allow the genetic algorithm to explore in many directions the state space.
- Values for the genetic operators such as population size and crossover and mutation probabilities.
- A scheduler to convert the generated chromosomes into schedules.

A. Chromosome Encoding

Two types of encoding can be used: the first is where the schedule is directly encoded in the chromosome; whereas, the second uses a scheduler to convert the encoded chromosomes into effective solutions. A critical decision is to define the elementary gene of a chromosome. This decision affects the number of eligible chromosomes to be generated as well as the level of difficulty of converting the chromosomes into real schedules.

The developed genetic algorithm is based on an indirect encoding. A chromosome is formed by genes representing the scheduling priorities of the jobs over the machines. As an example, consider a flow shop problem with 4 jobs. The chromosome (J_1, J_3, J_4, J_2) shows a preferable processing order of the jobs

B. Initial Population

The initial population can be generated by random process, duplication and evaluation of chromosomes or using a heuristic. For the proposed solution approach (GA_2), we used a random generation of chromosomes. This choice is based on the fact that the encoding used allows an easy generation of a large population of chromosomes.

C. Fitness Function

In order to evaluate the chromosomes we defined the following fitness function: $F(x) = MC - f(x)$, where MC represents the maximum value of f (objective function defined above) overall the genetic algorithm populations. A scheduler was used to convert every chromosome into a real schedule and returns its fitness function.

D. Genetic Operators

Selection: we used the lottery-wheel method described in [11] to select the chromosomes for the next generation. This method assigns a weight to every chromosome reflecting its contribution value to the population total fitness.

Crossover: Several crossover operators were published in the literature. We use the 1.X crossover proposed in [12]. The principle of this crossover is the following: Given two parents P1 and P2 and a random cut- point p, two children C1 and C2 are generated. Child C1 inherits the first p genes from the parent P1. Likewise, child C2 inherits the first p genes from the parent P2. The remaining genes of C1 are to be completed according to the missing genes from P2 whereas the missing genes of C2 are to be completed from P1 avoiding redundancies

Mutation: The mutation is a slight change of few randomly selected chromosomes. This is achieved in general by swapping some genes or inversing their values. The mutation allows the genetic algorithm to search for feasible solutions in many directions. For (GA_2) we used a simple mutation method which consists of exchanging two randomly-selected genes.

E. The Scheduler

The use of an indirect encoding requires a scheduler to convert each chromosome into a schedule and then provides its fitness value. The evaluation is obtained by a step-by-step construction of a solution through the information extracted from the genes and the maintenance data. Our scheduler works as follows:

We assign to each maintenance activity o_{kj} to be processed at time t on the machine M_j the following elements:

- An earliest start time $r_{kj} = S_{(k-1)j} + D_j + T^j_{\min} - \Delta t$
- A tardiest end time $d_{kj} = S_{(k-1)j} + T^j_{\max} + 2D_j$
- A priority at a time t $\left(t - r_{kj} \Big/ d_{kj} - D_j - r_{kj}\right)^2$.

Higher priorities are assigned to the delayed maintenance activities.

Given a chromosome, we progressively construct an active or a non-delayed schedule. Each time, we compare the priority of the current job (divided by n+1) with the priority of the maintenance activity. The task having the highest priority is placed first. Finally, we keep the best obtained solution among the active and the non-delayed schedules.

IV. Computational Results

To test the genetic algorithm, we used the following procedure to generate the maintenance and production data. This procedure is adapted from [13] and [14].

- The number of jobs n and machines m are as follows: $n \in \{20, \ldots, 100\}$ and $m \in \{2, 4, 6, 8\}$
- The processing times p_{ij} are uniformly distributed between 1 and 100.

- The earliest start times r_i are uniformly distributed between 0 and $\sum_{j=1}^{m} p_{ij}$
- The maintenance periods are :

$$T_j = \beta * \sum_{i=1}^{n} (p_{ij} + \sum_{k=1}^{j} \min_i p_{ik}) \Big/ m. \quad (4)$$

where $\beta \in \{0.2\ ,\ 0.4\ ,\ 0.8\}$.

- The maintenance processing times are:

$$D_j = \alpha * \sum_{i=1}^{n} p_{ij} \Big/ n\ . \quad (5)$$

where $\alpha \in \{1, 2, 3\}$

- The jobs' due dates d_i are uniformly distributed between $S*(1-\tau-R/2)$ and $S*(1-\tau+R/2)$, where:

$$S = \max\left[\max_{1\le j\le m}\left\{\sum_{i=1}^{n} p_{ij} + \min_i \sum_{l=j+1}^{m} p_{il} + D_j * \sum_{i=1}^{n} p_{ij}/T_j\right\}, \max_i\left\{\sum_{j=i}^{m} p_{ij} + D_j * \sum_{i=1}^{n} p_{ij}/T_j\right\}\right] \quad (6)$$

with R and τ are two parameters introduced to adjust the dispersion of the due date range and the tardiness factor. R and τ take their values respectively in $\{0.6, 1.2\}$ and $\{0.2, 0.4\}$

- The allowable earliness/tardiness times are $\Delta T_j = 0.05 * T_j$.

The experimental results of the proposed genetic algorithm (GA_2), were compared with those given by another genetic algorithm (GA_1) proposed in [15] which uses a direct machine-based encoding. Table I summarizes the details of computational results applied to 16 different sized problems categorized in 4 different classes C1 to C4 where:

- $C1: \{\tau = 0.2\ ,\ R = 0.6\}$,
- $C2: \{\tau = 0.2\ ,\ R = 1.2\}$,
- $C3: \{\tau = 0.4\ ,\ R = 0.6\}$,
- $C4: \{\tau = 0.4\ ,\ R = 1.2\}$.

Each algorithm was run 10 independent times for each problem instance. Table I shows the best values of the objective function f obtained by the two genetic algorithms GA_1 and GA_2 overall the runs. CPU_1sec and CPU_2sec represent the average running time of GA_1 and GA_2 under a Pentium (R) 4 CPU 2.80 GHZ.

According to Table I, we deduce that GA_2 outperforms GA_1 for all the problem instances in both objective function and running time. For instance, the performance of the new algorithm is much better mainly for big sized problems. In fact, the number of maintenance activities to be processed is proportionally increasing with the problem size. The encoding type used in the new genetic algorithm based on the priorities between the tasks is helping to schedule in a better way the jobs and the maintenance activities.

V. Conclusion and Further Research

We studied the flow shop scheduling problem under periodic maintenance activities. A new aggregated objective function including the production and maintenance criteria was developed. Both criteria are based on total earliness/tardiness times. To solve the problem, we proposed a genetic algorithm using an indirect encoding joined with a scheduler based on priority rules to convert the chromosomes into real schedules. The new algorithm was then tested, compared and validated on different sized problems. The experimental results clearly showed that both the objective function and the running time were improved compared to another genetic algorithm.

TABLE I
Results for 10 Independent Runs of GA_1 and GA_2

Problem	n	m	$f(GA_1)$	$f(GA_2)$	CPU_1sec	CPU_2sec
PB1-C1	20	2	90	85	19	10
PB2-C2	20	2	760	752	19	10
PB3-C3	20	2	3676	3420	19.5	15.2
PB4-C4	20	2	731	699	19.3	16.1
PB5-C1	40	4	4488	3520	48.5	33.2
PB6-C2	40	4	2750	2025	48.3	31.5
PB7-C3	40	4	11486	9520	48.5	30.02
PB8-C4	40	4	2544	1674	48.8	31.8
PB9-C1	60	6	8432	6230	95.3	74.1
PB10-C2	60	6	6061	5102	96	71.3
PB11-C3	60	6	40656	32140	94.9	72.03
PB12-C4	60	6	26650	22189	93.5	71.4
PB13-C1	80	8	18254	13258	168.5	95.6
PB14-C2	80	8	11566	9875	165.1	94.25
PB15-C3	80	8	78317	61250	167.1	91.47
PB16-C4	80	8	69153	51230	167.6	92.03

More realistic constraints such as random machine breakdowns can be considered in future studies. In addition, the objective function can be adjusted by including two parameters reflecting the weights of production and maintenance criteria. Another alternative for production and maintenance criteria is to use the Pareto optimality concept which allows generating a wide range of optimal solutions for each criterion separately. The need for an expert in this case will be a must to select the most appropriate solutions.

References

[1] M. Ben Ali, M. Sassi, M. Gossa, and Y. Harrath, "Simultaneous scheduling of production and maintenance tasks in the job shop", *International Journal of Production Research*, vol. 49, pp. 3891-3918, 2011.

[2] M. Kubzin, C. Potts, and V. Strusevich, "Approximation results for flow shop scheduling problems with machine availability constraints," *Computers & Operations Research*, vol 36, pp. 379-390, 2009.

[3] T. C. E. Cheng, and G. Wang, "An improved heuristic for the two-machine flowshop scheduling with an availability constraint", *Operations Research Letters*, vol 26, pp. 223-229, 2000.

[4] C.-Y. Lee, "Minimizing the makespan in the two-machine flowshop scheduling problem with an availability constraint," *Operations. Research. Letter*, vol 20, pp. 129–139, 1997.

[5] F. Hnaien, F. Yalaoui, and A. Mhadhbi, "Makespan minimization on a two-machine flowshop with an availability constraint on the first machine," *International Journal of Production Economics*, vol 164, pp. 95-104, 2015.

[6] J. Blazewicz, J. Breit, P. Formanowicz, W. Kubiak, and G. Schmidt, "Heuristic algorithms for the two-machine flow shop with limited machine availability," *The International Journal of Management Science*, vol 29, pp. 599-608, 2001.
[7] W. Kubiak, J. Blazewicz, P. Formanowicz, J. Breit, and G. Schmidt, 2002. "Two-machine flow shops with limited machine availability," *European Journal of Operational Research* vol 136, pp. 528-540, 2002.
[8] X. Wang, and J. Xie, "Two-stage flexible flow shop scheduling with limited machine availability," *First Int. Conf. on Information and Management Sciences*, Xi'an, China, 2002.
[9] R. Aggoune, "Ordonnancement d'ateliers sous contraintes de disponibilité des machines," Ph.D diss., University of Metz, France, 2002.
[10] B. Naderi, M. Zandieh, and M. Aminnayeri, "Incorporating periodic preventive maintenance into flexible flowshop scheduling problems", *Applied Soft Computing*, vol 11, pp. 2094-2101, 2011.
[11] D. Golberg, *Genetic algorithms in search, optimization and machine learning. Inc*, 1989.
[12] L. Davis, "Job shop scheduling with genetic algorithm," *1st Int. Conf. on Genetic Algorithms and their applications*, Lawrence Erlbaum (ED.), Hillsdale, New Jersey, 1985.
[13] C. Y. Lee and Z-L. Chen, "Scheduling jobs and maintenance activities on parallel machines," *Naval Research Logistics*, vol 47, pp. 145-165, 2000.
[14] Y. Xu, Y. Tian, and N. Sannomiya, "Three-stage tabu search for solving large-scale flow shop scheduling problems," *Transactions of Institute of Electrical Engineers of Japan*, vol 3, pp. 601-608, 2003.
[15] J. Kaabi, C. Varnier, and N. Zerhouni, 2004. "Ordonnancement conjoint de la production et de la maintenance dans un flow shop", 5ème conférence francophone de Modélisation et Simulation, Nantes, France, 2004.