# FMS JOB-SHOP SCHEDULING AND RESCHEDULING

LI ZHU*, YENG CHAI SOH and JINYAN MENG

School of Electrical and Electronic Engineering
Nanyang Technological University
Nanyang Avenue 639798
SINGAPORE

Abstract: - Flexible Manufacturing Systems (FMSs) have been developed over the last two decades to help manufacturing industries move towards the goal of flexibility. Scheduling is to optimize the use of resources so that the overall production objectives are met. In this paper, we present a two-stage procedure off-line scheduling algorithm based on the Lagrangian Relaxation method to maintain minimum mean weighted flow time for solving FMS job shop problems. Therefore, the FMS job shop problem is decomposed into machine-level sub-problems so that computation time is reduced significantly. On the other hand, uncertainties in the production environment inevitably result in deviations from the initial schedules. This makes rescheduling particularly important in real-time and dynamic environments in modern manufacturing systems. Four types of interruptions, i.e. machine breakdown, increased order priority, arrival of rush orders, and order cancellation, are considered in the case of rescheduling with an objective of maximum stability.

Key-Words: - FMS, job shop, scheduling, rescheduling, Lagrangian relaxation, interruptions

## 1. INTRODUCTION

In an Flexible manufacturing system (FMS), scheduling is the process of organizing, choosing and timing resource usage to carry out all the activities necessary to produce the desired ouputs at the activities and the resources[1]. The development of effective and efficient FMS scheduling strategies remains an important and active research area. However, scheduling in FMSs differs from that in a conventional job shop because of the availability of alternative manufacturing resources resulting in routing flexibility. On the other hand, there are always random and unpredictable events in the system, such as material shortage, operator absence or machine breakdown, due date change, etc. Then the generation of new and modified schedule, i.e. rescheduling, is needed when such dynamic events occur. This is becoming essential in today's complex manufacturing environment.

Scheduling methodologies based on Lagrangian relaxation have been used to improve the computation efficiency with near-optimal solutions [2] [3] for decades. The Lagrangian multipliers act as prices to regulate the use of machines. However, the solution of the dual problem is not in general feasible due to the non-convexity of the scheduling problem or the stopping criterion used, thus a good feasible constructing method is necessary. In [4], an augmented Lagrangian relaxation approach was presented to decompose job shop scheduling problems with routing consideration into operation-level subproblems. The Gauss-Seidel iterative approach is adopted to solve a cross term involving the beginning times of the current operation and the following operation. The computational complexity mainly depends on the time horizon and the Gauss-Seidel iterations. If the time horizon is large, the computation time is long but still acceptable.

In practise, a shop floor is seldom stable for more than half an hour[5]. Noting that the capacity lost via disruption on a fully loaded machine ia not able to be recovered, Yellig and Mackulak [6] propose an strategy, called capacity hedge, to alleviate scheduling nervousness under disruptions if the system has spare capacities. In this approach, capacity is held in reserve to protect from a stochastic failure event. It combines reliability characteristics [7] [8] with the anticipatory failure policy of inventory hedge points [9].

This paper focuses on scheduling and rescheduling problems using Lagrangian relaxation and Affected Operation method. First, a two-stage approach to solve the decomposed problem is presented. Second, the FMS job shop rescheduling algorithm under random disruption with stability(i.e. deviation from the initial schedule) objective is presented. Finally, numerical results obtained from the proposed approaches are presented to illustrate that the method can obtain satisfactory solutions.

## 2. LAGRANGIAN RELAXATION METHODS FOR OFF-LINE SCHEDULING IN FMS

In general, job shop scheduling can be formulated as an optimisation problem which has objective functions with equality and inequality constraints. Consider a FMS which has $M$ machines. Each machine can process a set of operations. Different machines manage different operation sets, and have different processing time of the same opration. The job shop is fed by $N$ jobs belonging to $K$ job classes.

Each job has several operations. The process order of operations of a job is fixed, and different job classes include different operations. Each job also has sequence-dependent set-up time according to which machine it is processed and is assigned a weight(due date) that varies among different job classes. Our off-line scheduling generates a sequence of jobs on each machine before any of the operations begins. The precedence constraints of operations belonging to the same job, machine capacities and multiple routes of operations are taken into consideration to achieve minimum weighted flow time of jobs in the system. Different from [4] [10] which relaxed the precedence constraints and machine capacity constraints, our scheduling approach based on Lagrangian relaxation method is presented with a different mathematical model and a different relaxation scheme.

The definitions of variables and parameters are given in Table 1. $C_i$ represents the completion time of the last operation of job $i$, that is $C_i = c_{iN_i}$. All jobs are assumed to be available for processing at time 0. The time horizon $T$ is assumed to be long enough to compete all the jobs, and $H \geq T$.

| | |
|---|---|
| $C_i$ | completion time or flow time of job $i$ |
| $c_{ij}$ | completion time of operation $(i,j)$ |
| $H$ | an arbitrarily large number |
| $(i,j)$ | operation $j$ of job $i$ |
| $M$ | the number of machines |
| $N$ | the number of jobs |
| $N_i$ | the number of operations of job $i$ |
| $O_m$ | the set of operations that can be operated on machine $m$ |
| $p_{ijm}$ | the processing time of operation $(i,j)$ on machine $m$ |
| $T$ | time horizon under consideration |
| $v_{ijm}$ | $=\begin{cases} 1 & \text{if operation } (i,j) \text{ is scheduled on machine } m \\ 0 & \text{otherwise} \end{cases}$ |
| $\omega_i$ | weight(value of importance) of job $i$ |
| $y_{iji'j'm}$ | $=\begin{cases} 1 & \text{if } v_{ijm}=0 \text{ or } (i,j) \text{ procedes } (i',j') \text{ on machine } m \\ 0 & \text{otherwise} \end{cases}$ |
| $Z_{i,j}$ | the set of alternative machine options for operation $(i,j)$ |

Table 1: Definition of Variables and Parameters

If the cost function is minmized by the mean weighted flow time of the jobs, then the problem can be formulated as follows:

$$\text{Problem} \quad P: \quad \min \sum_{i=1}^{N} \omega_i c_{iN_i} \tag{1}$$

subject to:

$$\begin{array}{l} c_{ij+1} - c_{ij} \geq \quad p_{ij+1m} v_{ij+1m}; \\ (i=1,\ldots,N, \quad j=1,\ldots,N_i-1, \quad m \in Z_{ij+1}) \end{array} \tag{2}$$

$$\begin{array}{rl} c_{ij}v_{ijm} - c_{i'j'}v_{i'j'm} + Hy_{iji'j'm} & \geq p_{ijm}v_{ijm} \\ c_{i'j'}v_{i'j'm} - c_{ij}v_{ijm} + H(1-y_{iji'j'm}) & \geq p_{i'j'm}v_{i'j'm} \\ m=1,\ldots,M, \quad (i,j),(i',j') \in O_m & \wedge (i,j) \neq (i',j') \end{array} \tag{3}$$

$$c_{i1} \geq p_{i1m} v_{i1m}; \qquad i=1,\ldots,N, \quad m \in Z_{i1} \tag{4}$$

$$\sum_{m \in Z_{ij}} v_{ijm} = 1; \qquad i=1,\ldots,N, \quad j=1,\ldots,N_i \tag{5}$$

The constraints(2)assure precedence relationships of operations belonging to the same jobs. Constraints (3) ensure no more than one operation are processed on any machine simultaneously. Constraints (4) assure that if machine $m$ is used for the first operation of a job, then the completion time of the operation must be equal to or greater than its processing time on that machine. Constraints (5) ensure only one machine is selected for an operation $(i,j)$.

Constraints(2) can be relaxed by using the nonnegative Lagrange multipliers $\lambda_{ijm}$. This leads to the folllowing relaxed problem:

$$\text{Problem} \quad LP: \quad \min_{c_{ij}, v_{ijm}} L \tag{6}$$

s.t.(3),(4),(5) and $\lambda_{ijm} \geq 0$ where

$$L = \sum_{i=1}^{N} \omega_i c_{iN_i}$$

$$+ \sum_{i=1}^{N} \sum_{j=1}^{N_i-1} \sum_{m \in Z_{ij+1}} \lambda_{ijm} [c_{ij} - c_{ij+1} + p_{ij+1m} v_{ij+1m}] \tag{7}$$

The dual problem to problem $LP$ is

$$\text{Problem} \quad DLP: \quad \max_{\lambda \geq 0} q(\lambda) \tag{8}$$

s.t. (3),(4),(5) and $\lambda_{ijm} \geq 0$ where

$$\begin{aligned} q(\lambda) &= \min_{c_{ij}, v_{ijm}} L \\ &= \min_{c_{ij}, v_{ijm}} \sum_{i=1}^{N} \{ \omega_i c_{iN_i} \\ &+ \sum_{j=1}^{N_i-1} \sum_{m \in Z_{ij+1}} \lambda_{ijm} (c_{ij} - c_{ij+1} + p_{ij+1m} v_{ij+1m}) \} \end{aligned} \tag{9}$$

The decomposed problem $LP$ is also complicated due to the routing constraints (5) of individual operations. Optimal solution can be obtained by an enumeration procedure, which is complex in computation. Here, a local variable $c_{ijm}$ is introduced as the completion time of operation $(i,j)$ on machine $m$. If operation $(i,j)$ is not processed on the machine $m$, $c_{ijm} = 0$. Constraints (5) insures that

$$c_{ij} = \sum_{m \in Z_{ij}} c_{ijm} \tag{10}$$

Substituting equation (10) into equation (7) becomes:

$$L = \sum_{m=1}^{M} \sum_{(i,j) \in O_m} (a_{ij} c_{ijm} + b_{ijm} v_{ijm}) \tag{11}$$

in which,

$$\begin{cases} a_{i1} &= \sum_{k\in Z_{i2}} \lambda_{i1k} \\ a_{ij} &= \sum_{k\in Z_{ij+1}} \lambda_{ijk} - \sum_{k\in Z_{ij}} \lambda_{ij-1k}; \\ & \qquad j = 2,\dots,N_i - 1 \\ a_{iN_i} &= \omega_i - \sum_{k\in Z_{iN_i}} \lambda_{iN_i-1k} \\ b_{i1m} &= 0 \\ b_{ijm} &= \lambda_{ij-1m}p_{ijm}; \quad j = 2,\dots,N_i \end{cases}$$

Problem $LP$ is equal to:

$$\min_{c_{ijk},v_{ijk}} L \tag{12}$$

$$s.t. \quad (3),\ (4)\ and\ (5).$$

## 2.1. First Stage: one-machine multi-operation subproblem

In this stage, constraints (5) are ignored, and an operation $(i,j)$ with $l$ altenatve machine options is treated as $l$ independent operations. That is, let $v_{ijm} = 1$ for $m \in Z_{ij}$. Then problem $LP$ can be readily decomposed into $M$ independent subproblems. For machine $m$, the subproblem is:

$$\text{Problem} \quad LPm: \quad \min_{c_{ijm}} L_m \tag{13}$$

in which,

$$L_m = \sum_{(i,j)\in O_m} (a_{ij}c_{ijm} + b_{ijm}v_{ijm}) \tag{14}$$

s.t.

$$\left.\begin{array}{lcl} c_{ijm} - c_{i'j'm} + Hy_{iji'j'm} & \geq & p_{ijm} \\ c_{i'j'm} - c_{ijm} + H(1 - y_{iji'j'm}) & \geq & p_{i'j'm} \end{array}\right\};$$

$$(i,j),(i',j') \in O_m \ \ and \ \ (i,j) \neq (i',j') \tag{15}$$

$$c_{i1m} \ \geq \ p_{i1m}; \quad (i,1) \in O_m \tag{16}$$

Hence, $LPm$ is a kind of one-machime multi-job scheduling problems with the measurement of mean weighted flow time. The operations can be divided into three sets, $O_{mp}$, $O_{m0}$ and $O_{mn}$, based on that the operation weight $a_{ij}$ is positive, zero or negative accordingly.

The optimal schedule is obtained if the operations are scheduled in the following manner:

$$\frac{p_1}{a_1} \leq \frac{p_2}{a_2} \leq \dots \leq \frac{p_l}{a_l} \leq \dots \leq \frac{p_{n_p}}{a_{n_p}}; \quad l \in O_{mp} \tag{17}$$

$$p_1 \leq p_2 \leq \dots \leq p_l \leq \dots \leq p_{n_0}; \quad l \in O_{m0} \tag{18}$$

$$\frac{p_1}{a_1} \geq \frac{p_2}{a_2} \geq \dots \geq \frac{p_l}{a_l} \geq \dots \geq \frac{p_{n_p}}{a_{n_p}}; \quad l \in O_{mn} \tag{19}$$

where $n_p$ $n_0$, and $n_n$ are the number of operations of set $O_{mp}$, $O_{m0}$ and $O_{mn}$ respectively, $p_l$ is the processing time and $a_l$ is the weight of operation $l$.

Using above approach, the value of completion time for every operation on any route can be assigned.

## 2.2. Second Stage: selecting the best operation route

A cost change function is defined as:

$$f_m(i,j) = f_{mog}(i,j) - f_{mcg}(i,j) = a_{ij}c_{ijm} + b_{ijm}v_{ijm} \tag{20}$$

where $f_{mog}(i,j)$ and $f_{mcg}(i,j)$ is the costs obtained from equation (14) when the alternative route for operation $(i,j)$ on machine $m$ is not deleted and deleted, respectively. The route with the smallest cost change, $f_m(i,j)$ $(m \in Z_{ij})$, should be selected. If two routes have the same value of cost change, choose one with less load, i.e. the machine with minimum $\sum_{(i,j)\in O_m} p_{ijm}$.

## 2.3. Solving the Dual Problem

Although it is a near optimal cost of problem $LP$, $\tilde{L}$ is used to maximize the Lagrangian dual cost $\tilde{q}$ following (9):

$$D\tilde{L}P: \qquad \max_{\lambda\geq 0} \tilde{q}(\lambda) \tag{21}$$

$$\text{with} \qquad \tilde{q}(\lambda) = \tilde{L} \tag{22}$$

To solve the dual problem $D\tilde{L}P$, the common subgradient method is used to update the multiplier values $\lambda$.

$$\lambda^{n+1} = \lambda^n + \alpha^n g(\lambda^n) \tag{23}$$

and

$$\alpha^n = \beta \frac{q^u - \tilde{q}^n}{g(\lambda^n)^T g(\lambda^n)} \tag{24}$$

where $\beta$ is the step size at $n$th iteration, and $g(\lambda^n)$ is the subgradient of $D\tilde{L}P$. The elements in $g(\lambda^n)$ are given by:

$$g_{ij} = p_{ijm}v_{ijm} + \quad c_{ij} - c_{ij+1}; \\ i = 1,\dots,N, \quad j = 1,\dots,N_i - 1,\ m \in Z_{ij+1} \tag{25}$$

In equation (24), $q^u$ is an estimate of the optimal solution of (22), and $\tilde{q^n}$ is the value of $\tilde{q}$ at the $n$th iteration.

## 2.4. Constructing of a Feasible Schedule

Since only the precedence constraints are relaxed, the relaxed solution may be not feasible. Hence, after sovling the dual problem, the sequence of operations on any one of machines is rearranged in order to ensure that the processing sequence of a job satisfies the operation precedence requirements of the job.

## 2.5. Procedure of Scheduling Algorithm

For the approach given here, the solution obtained may not be an optimal solution of the relaxed problem $LP$. So, in every iteration, a feasible solution is constructed. Then the best one among all the feasible ones is chosen as the final solution of the scheduling problem.

We note that the two-stage approach involve heavy computation due to considering all possible permutations for all operations with alternative machine options. However, it seems impractical to use standard Lagrangian relaxation method to the scheduling problems raised in this paper. This has been is pointed out by [4].

## 3. RESCHEDULING UNDER RANDOM DISRUPTIONS

It is well known that a shop floor may not be a static environment. Previous research concentrated on developing optimal scheduling heuristics and algorithms under a stable, deterministic shop floor environment. Here, we consider rescheduling problems.

Ramasesh [11] presented a survey involving dynamic job shops scheduling. Matsuura et al.[12] studied rescheduling problem using an approach of selecting between sequencing and dispatching in case of uncertainties, which was carried out by simulation. Li et al.[13] proposed a rescheduling algorithm based on the construction of a scheduling binary tree and a net change concept adopted from MRP systems. However, all above studies did not consider the alternate routeings for rescheduling, i.e. they assumed no change in the existing operation sequence for each machine.

In this paper, the rescheduling disruption is focused on the following five different types of uncertainties:

- machine breakdown
- increased order priority (i.e. the change in due dates)
- the arrival of rush orders
- order cancellation
- delays in the arrival of materials
  When the arrival of materials for a operation is delayed, all the remaining jobs should be remembered, then, deleted as order cancellation and added as a rush order when the materials are available again.

Here, we proposed rescheduling mechanisms on the former four types of disruption problems. Rescheduling commences from the time a disturbance occurs and takes into account the current state of production on the shop floor. Factors such as alternative machine, sequence dependent setup time, and idle time on machine are taken into consideration. Hence, sequence deviation exists when changeover succeeds. While on the other hand, the increase of makespan is reduced by this method.

### 3.1. Rescheduling mechanism for Machine Breakdown

When a machine is breakdown, the remaining operations of the job may have to be performed using other machines. Hence, the idle time on DownM [1], the setup time of broken operation on AM, the priority of the broken operation are three important factors in rescheduling. The rescheduling procedure is shown as follows:

- step 1: Find the broken machine(DownM) and interrupted operation, assign expected downtime for this machine, go to step 2.
- step 2: If there is any operation currently on DownM, revise the operation status(time remaining), go to step 3. Else, go to step 10.
- step 3: If idle time on DownM less than downtime, go to step. Else, go to step 10.
- step 4: If there are alternative machines(AM) available, check setup and processing time required for all alternative ones. Chose the least utilized machine, go to step 5. Else, go to step 10.
- step 5: If setup time on AM is less than the downtime of the broken machine, go to step 6. Else, go to step 10.
- step 6: If AM is free, go to step 7. Else, go to step 8.
- step 7: If the idle time on AM is more than the setup time on AM, or broken operation priority is higher than the operation currently performed on AM, assign broken operation to it and update the system status. Else, go to step 10.
- step 8: If broken operation priority is higher than the operation currently performed on AM, then preempt the alternative machine and update the system status. Else, go to step 9.
- step 9: If the difference between the ready time of the DownM and release time of AM is more than the setup time on AM, then assign broken task to alternate machine when it becomes free. Else, go to step 10.
- step 10: Interrupted operation wait on the same machine till it is ready, go to step 11.
- step 11: Update the machine status list.

### 3.2. Rescheduling mechanism for Increased Priority

Likewise, when a job priority is increased dynamically, rescheduling is accomplished by bringing its remaining tasks forward in time on using the following procedures:

- step 1: Find the job whose priority is increased, assign the highest priority to all the operations belonging to this job, revise the operation status, go to step 2.
- step 2: Find the first increased priority operation which is currently not loaded on any machine while going to be processed next, go to step 3.
- step 3: If the machine required by the increased priority operation is free at time T*[2], assign task to the machine, go to step 6. Else, go to step 4.
- step 4: If there are alternative machines(AM) available, check the status of the least utilized one. If it is free, assign the increased priority operation to the machine at time T*, go to step 6. Else, go to step 5.
- step 5: Pre-empt the initial machine and start high priority operation immediately and update the system status, go to step 6.
- step 6: Advance all the remaining operations to start immediately, broken operation waits on the same machine until increased priority operation is completed, go to step 6.
- step 6:Revise system status.

[1] It is the difference between the OprStart of nom of broken operation and the OprEnd of broken operation.

[2] T* is the OprEnd if there is an incresed priority operation currently loaded. Otherwise, it is the current time.

3.3. Rescheduling mechanism for Arrival of Rush Order

When a new order arrives, if it is important or rush, the highest priority is assigned to it; otherwise based on standard priority rules such as FCFS or EDD. In the former condition, all the least utilized machines required by the rush order are released whenever they are required. The algorithm for rush order arrival is almost the same as the algorithm for increased order priority except that there is an increase in the number of jobs flowing through the system.

3.4. Rescheduling mechanism for Order Cancellation

When an order is cancelled, all the remaining tasks of that order are deleted from the task list and the same time the system updates the time and machine status of the remaining jobs.

3.5. Affected Operations Rescheduling

After update the status of machines and uncompleted operations by using algorithms described above, Affected Operations Rescheduling(AOR) method is used to obtain the new schedule[14].

One of the objectives of rescheduling is to minimize the deviation. So, the performance is measured according to the efficiency (e.g. makespan) and stability (i.e. deviation from the initial schedule). The latter including two measures:

- $TimeDeviation(devT)$: there exist both $delay$ and $rush$ which stand for the absolute value of positive and negative differences in ending times.

$$devT = delay + rush$$

- $SequenceDeviation(devS)$: this measure is critical if setups are prepared in advance, i.e. jobs may wait on pallets in a sequence queue, and tooling and fixturing may be planned according to the original sequence. Thus, resequencing the queue, reallocating the pallets, and replanning the tools will incur costs[15]. Here we employed the method presented by Watatani and Fujii[16]:

$$Sequencedeviation(devS) = \sum_k \sum_j Seq_{jk}$$

4. NUMERICAL RESULTS

Two Example problems with the machine and operation parameters shown in Table 2 and Table 3 were generated for illustrating the proposed scheduling and rescheduling algorithms. A manufacturing system with five machines, which are capable of processing different jobs, is considered. Each job has several operations that should be performed in a strict sequential manner. Priorities(or weights ) among the jobs are assigned with respect to their due dates.

Here, the initial schedule is obtained by using Lagrangian Relaxation method discribed in section 2. The initial multipliers are set to zero. A feasible solution is caculated based on the dual problem and the best solution is conserved as the final solution. In the second part of the research, new schedule is generated by using rescheduling algorithms discussed in section 3. We use the system status as input and reschedule the operations when disturbance occur.

| Job (i) | Opr (i,j) | Machine Type | | | | | Wi |
|---|---|---|---|---|---|---|---|
| | | M1 | M2 | M3 | M4 | M5 | |
| 1 | 11 | - | 10/20 | 12/21 | - | - | 4 |
| | 12 | - | - | - | 8/22 | 12/20 | |
| | 13 | 5/15 | - | 6/12 | - | - | |
| 2 | 21 | - | - | - | - | 8/14 | 3 |
| | 22 | 10/18 | 8/22 | | - | - | |
| | 23 | 7/29 | - | 10/30 | - | - | |
| 3 | 31 | 12/38 | 12/40 | - | - | 14/36 | 2 |
| | 32 | - | - | - | 9/21 | 7/23 | |
| | 33 | 23/41 | - | - | - | 21/39 | |
| 4 | 41 | 21/53 | - | 18/52 | 22/50 | - | 1 |
| | 42 | - | 9/33 | - | 12/28 | - | |
| | 43 | 6/22 | 8/23 | - | - | - | |

Table 2: Setup/Processing time requirements (Example 1)

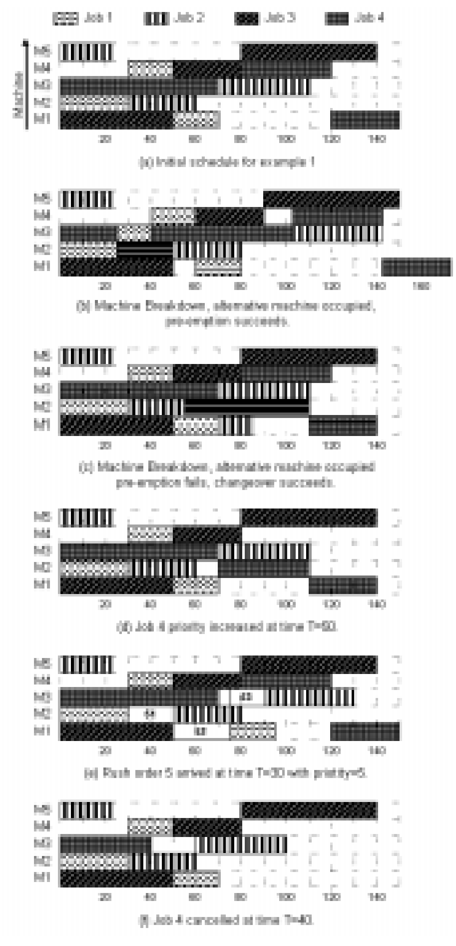


Figure 1: Rescheduling under disrruptions.

Some of rescheduling results of the Example 1 is shown in Gantt Charts[3] Fig.1. The charts graphically display the use of each resource at all times. The $x$ axis represents time and the $y$ axis consists of a horizontal bar for each machine or job. Numbers inside the blocks stand for the operation associated with the jobs, and black rectangles represent the machine failure periods.

In Fig. 1b, alternative machine M3 is found for operation 11 when machine 1 breaks down at T = 25. Since the broken operation 11 has a higher priority than that of task 41 which is processing on M3, task 41 is splited and starts on the same machine after operation 11 is completed. Here, the setup for task 41 on M3 is required due to tool changing between different operations. In Fig. 1c, machine breakdown happens at time T = 55 for 55 time units. Because the difference between the readyTime of M2 and OprEnd of operation 13is more than the setup time of task 22 on M1, changeover succeeds without pre-emption, i.e. broken operation 22 processes on M1 after task 13 finishes. Fig. 1d shows the priority of job 4 is increased at time T = 50. Operation 41 is currently loaded. Since M2 which is required by operation 42 is not free at OprEnd of task 41, the system is checked for alternative machines. M2, the free AM, is selected to process task 42 right after operation 41 completes. Then the remaining operation 43 of the increased priority job is advanced to start as soon as task 42 finishes. This results in earlier completion time of job 4. In Fig. 1e, rush job 5 arrives at T =30. Since its priority is 5, highest one, all the machines required by the rush order are released whenever they are required. Fig. 1f shows the effect of order cancesllation on the schedule. At time T = 50, job 4 is cancelled, which advances operation 23 forward. Also, completion of job 2 becomes earlier.

Among the four types interruptions discussed in this paper, machine breakdown is the most complicated one. Hence, we do some more simulations on it. Both short and long machine breakdowns are tested on each machine along the time axis. Here, we use 5 as short downtime and 25 as long downtime in Example 1 with consideration of the average setup time on machines. In Example 2, 4 and 20 are tested as short and long downtime respectively. Simulation results are shown in Fig. 2, Fig. 3, Fig. 4 and Fig. 5.

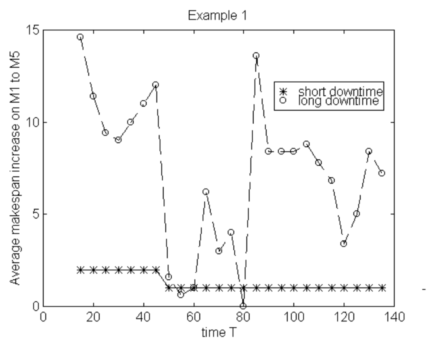


Figure 2: Average makespan increase of Example 1.

Fig. 2 and Fig. 3 illustrate the average makespan increasement from M1 to M5 when rescheduling under machine breakdowns, while Fig. 4 and Fig. 5 illustrate the average devT from M1 to M5 between new schedule and initial one. The simulation results show that long downtime have more impact on makespan increase. And the effects of early breakdowns are more distinct than late breakdowns because there is less idle time on machines. Time deviation(devT) is obvious when pre-emption succeeds.

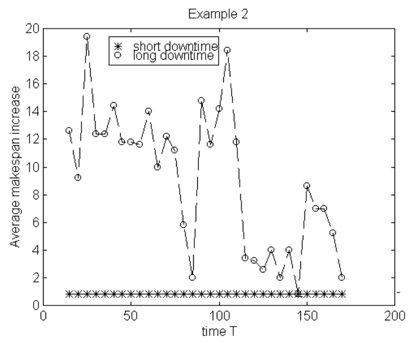


Figure 3: Average makespan increase of Example 2.

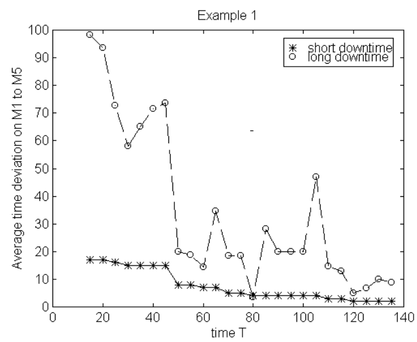


Figure 4: Average TimeDeviation of Example 1.

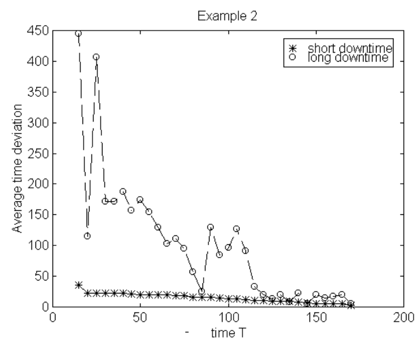


Figure 5: Average TimeDeviation of Example 2.

The sequence deviation(devS) is zero in most occasions because it reschedules by preserving most of the machine sequences except that the broken operation may changeover to alternative machine. In this way, the robustness of the initial schedule is obtained. Moreover, the results of the new schedule is improved compared with the method used by Abumaizar and Svestka[14] on makespan increase.

[3]Gantt Charts is named after Henry Gantt who developed the concept in the late 1800s.

## 5. CONCLUSIONS

First, An approach based on Lagrangian relaxation method has been proposed to solve job shop FMS scheduling problems. By relaxing the precedence constraints of the operations, the original problem is decomposed as one-machine multi-operation scheduling problems. Then we present a two-stage approach, with the measurement of minimizing the weighted flow time of jobs, to solve the decomposed problems by which it is easy to get a near-optimal solution.

Secondly, we proposed an affected operations rescheduling algorithm with a goal of scheduling stability in dynamic environment. Those operations which are to be scheduled at the time of interruption are rescheduled. It can solve random disruptions such as machine breakdown, job priority increasement, arrival of rush order and order cancellation, etc., with consideration of sequence dependent setup time. The numerical results show that our rescheduling can obtain satisfactory solution. Since alternative machine choices are available for the operations in FMS, the effect of interruptions on the system's performance is minimized.

| Job (i) | Opr (i,j) | Machine Type | | | | | Wi |
|---|---|---|---|---|---|---|---|
| | | M1 | M2 | M3 | M4 | M5 | |
| | 11 | 5/15 | - | 7/18 | - | - | |
| 1 | 12 | - | 8/30 | - | 12/28 | 10/32 | 6 |
| | 13 | 5/15 | - | 4/18 | - | - | |
| | 14 | - | 5/12 | - | - | 8/12 | |
| | 21 | - | - | - | 10/40 | - | |
| 2 | 22 | 7/18 | 5/15 | | - | 8/14 | 5 |
| | 23 | - | - | 8/32 | 5/30 | - | |
| | 31 | 10/20 | 12/20 | - | - | 8/20 | |
| | 32 | - | - | - | 4/16 | 5/20 | |
| 3 | 33 | 8/36 | - | 8/32 | - | - | 4 |
| | 34 | - | 6/17 | - | 5/20 | - | |
| | 35 | 6/12 | - | 4/12 | - | - | |
| | 41 | - | - | 5/12 | - | - | |
| | 42 | - | 12/20 | - | 8/22 | 10/20 | |
| 4 | 43 | 15/33 | - | 10/40 | - | - | 4 |
| | 44 | - | - | - | 5/15 | 8/15 | |
| | 45 | 5/20 | 6/14 | - | - | 4/16 | |
| | 51 | - | 10/30 | - | - | 13/25 | |
| 5 | 52 | - | 7/20 | 5/25 | - | - | 3 |
| | 53 | - | - | - | 12/35 | - | |
| | 61 | 7/18 | 5/15 | - | - | 7/15 | |
| 6 | 62 | - | - | 8/25 | 7/28 | - | 2 |
| | 63 | - | 6/19 | - | 6/22 | - | |
| | 64 | 4/14 | - | 5/15 | - | 6/12 | |
| | 71 | 8/17 | - | - | 5/15 | - | |
| 7 | 72 | 5/17 | - | 7/20 | - | - | 2 |
| | 73 | - | 4/14 | - | 7/15 | 3/15 | |
| | 81 | - | - | 10/40 | 13/35 | - | |
| 8 | 82 | 7/13 | - | - | 5/20 | 6/16 | 1 |
| | 83 | - | 7/28 | 8/24 | 10/20 | - | |

Table 3: Setup/Processing time requirements of operations of Example 2

## 6. REFERENCES


[1] T. E. Morton and D. W. Pentico, Heuristic Scheduling Systems. New York: John Wiley and Sons, 1993.

[2] M. L. Fisher, "Optimal solution of scheduling problems using lagrange multipliers, part i," Operation Research, vol. 21, pp. 1114–1127, 1973.

[3] M. L. Fisher, "Lagrangian relaxation method for solving integer programming problems," Management Science, vol. 27, pp. 1–18, 1981.

[4] D. J. Hoitomt, P. B. Luh, and K. R. Pttipati, "A practical approach to job-shop scheduling problems," IEEE Transactions on Robotics and Automation, vol. 9, pp. 1–13, 1993.

[5] K. N. Mckay, f. R. Sfayeni, and J. A. Buzacott, "Job-shop scheduling theory: what is relevant?," Interfaces, vol. 18, pp. 84–90, 1988.

[6] E. Yellig and G. Mackulak, "Robust deterministic scheduling in stochastic environments: the method of capacity hedge points," International Journal of Production Research, vol. 35(2), pp. 369–379, 1997.

[7] M. Berg, M. J. M. Posner, and H. Zhao, "Production-inventory systems with unreliable machines," Operation Research, vol. 42, pp. 111–118, 1994.

[8] R. R. Meyer, M. H. Rothkopf, and S. A. Smith, "Reliability and inventory in a production-storage system," Management Science, vol. 25, pp. 799–807, 1979.

[9] J. Kimemia and S. B. Gershwin, "An algorithm for the computer control of a flexible manufacturing system," IIE Transactions, vol. 15, pp. 353–362, 1983.

[10] C. S. Czerwinski and P. B. Luh, "Scheduling products with bills of materials using an improved lagrangian relaxation technique," IEEE Transactions on Robotics and Automation, vol. 10(2), pp. 99–111, 1994.

[11] R. Ramasesh, "Dynamic job shop scheduling: a survey of simulation research," OMEGA International Journal of Management Science, vol. 18(1), pp. 43–97, 1990.

[12] H. Matsuura, H. Tsurone, and M. Kanezashi, "Sequencing dispatching and switching in a dynamic manufacturing environment," International Journal of Production Research, vol. 31(7), pp. 1671–1688, 1993.

[13] R. Li, Y. ShYu, and S. Adiga, "A heuristec rescheduling algorithm for computer-based production scheduling systems," International Journal of Production Research, vol. 31(8), pp. 1815–1826, 1993.

[14] R. J. Abumaizar and J. A. Svestka, "Rescheduling job shops under random disruptions," International Journal of Production Research, vol. 35(7), pp. 2065–2082, 1997.

[15] D. S. Wu, R. H. Storer, and P. C. Chang, "One machine recheduling heuristics with efficiency and stability as criteria," Computers and Operations Research, vol. 20(1), pp. 1–14, 1993.

[16] Y. Watatani and S. Fujii, "A study on rescheduling policy in production systems," in Proceedings of the JAPAN/USA Symposium on Flexible Automation, pp. 1147–1150, 1992.