



STUDY OF EVENT-DRIVEN AND PERIODIC RESCHEDULING ON A SINGLE MACHINE WITH UNEXPECTED DISRUPTIONS

*Moaz Magdy Tawfeek
Ain Shams University, Cairo, Egypt
Moazmagdy91@gmail.com*

*Yomna Mahmoud Sadek
Ain Shams University, Cairo, Egypt
yomna.sadek@eng.asu.edu.eg*

*Amin M. K. Elkharbotly
Ain Shams University, Cairo, Egypt
amin_elkharbotly@eng.asu.edu.eg*

Submission: 14/05/2018

Accept: 07/06/2018

ABSTRACT

This paper studies the rescheduling problem of a single machine facing unexpected disruptions in order to determine which parameters can help reducing the negative impacts of these disruptions on schedule performance. A Genetic Algorithm (GA) is used to generate the initial schedule and the updated ones according to a reactive strategy. The performance of event-driven rescheduling and periodic rescheduling policies are compared in terms of total tardiness and total cost of rescheduling. Other factors that may affect rescheduling such as disruption time, disruption duration and number of disruptions are investigated. The sensitivity of results to both due date tightness and cost factor variation is tested. The results showed that the timing of the occurrence of disruption as related to scheduling horizon has a major effect on determining the best rescheduling policy. Event-driven policy is superior to other policies for short infrequent disruptions. It was found that the periodic policy is more appropriate for long and frequent disruptions.

Keywords: Single machine, rescheduling, event-driven, periodic, rescheduling frequency



1. INTRODUCTION

In a production system, the production schedule is developed for a certain scheduling horizon and released to the shop floor before production begins. However, once a schedule is released to the shop floor, it is immediately subject to random disruptions, which may render the original schedule obsolete and necessitate rescheduling. Examples of unexpected disruptions “rescheduling factors” include machine breakdown, arrival of urgent jobs, cancellation of an order, operator absenteeism, rework or quality problems, and shortage of raw materials.

Rescheduling is the process of generating a new executable schedule in response to unexpected disruption. Because disruptions at the shop floor are inevitable, the importance of rescheduling is comparable with that of scheduling. Proper rescheduling requires generating a new efficient schedule in a timely manner.

The classification of rescheduling studies introduced by Vieira et al. (2003) is adopted in this study. They have classified the rescheduling studies into four categories according to the *rescheduling environment*, *rescheduling strategies*, *rescheduling methods*, and *rescheduling policies*. In their classification, the rescheduling environment identifies the set of jobs that need to be scheduled/rescheduled; it can be static or dynamic.

The rescheduling strategy describes whether initial production schedules are generated or the jobs will be dispatched dynamically. The rescheduling policy specifies when rescheduling should occur; it can be event-driven, periodic or hybrid. And rescheduling methods describe how updated schedules are generated; it can be complete schedule regeneration or only repairing the original schedule.

From an industrial point of view, the aim of rescheduling is typically to find a trade-off between efficiency in terms of a specified scheduling objective function (e.g. makespan, total flow time and total tardiness) and stability in terms of the difference between original and new schedules. The alternations after disruptions to the initial schedule inevitably make some earlier preparations (e.g. setups, materials supply and tooling) obsolete, thereby incurring additional costs. It is necessary and reasonable for the newly generated schedule to minimize or restrict the variances from the initial schedule as reported by LIU and ZHOU (2015).

The literature on production rescheduling includes different machine environments. MASON et. Al. (2004), DONG and JANG (2012), MORATORI and PETROVIC (2012) studied the rescheduling of job shops. While Caricato and Grieco (2008), Chiu and shih (2012) and Katragjini, Vallada and Ruiz (2013), studied the rescheduling of flowshop. Parallel machine rescheduling is studied by Vieira, Hermann and Lin (2000), Liu and Zhou (2013) and Arnaout (2014). Also Flexible manufacturing systems (FMS) rescheduling studies was found in literature such as Zakaria and Petrovic (2012) and Jain, Foley (2016).

Although several studies have considered the single machine rescheduling problem for new jobs arrival, little attention has been given to the rescheduling problem with machine breakdown.

The problem of new job insertion into a single machine schedule has been studied by LIU and ZHOU (2015), VIEIRA, HERMANN, and LIN (2000), LIU and ZHOU (2012), AKKAN (2015), HOOGEVEEN, LENTÉ, T'KINDT (2012), UNAL, UZSOY, and KIRAN (1997), YANG (2007) and GUO et al. (2016).

Vieira, Hermann and Lin (2000) suggested an analytical model, which uses FIFO dispatching rule to sequence jobs, to estimate the performance of both event-driven and periodic rescheduling in terms of average flow time and machine utilization. While Liu and Zhou (2012) used Variable Neighborhood Search (VNS) algorithm to solve to near optimal solution for single machine maximum lateness rescheduling problem subject to an upper limit on the total sequence disruption.

Akkan (2015) used simple heuristics in addition to hybrid branch-and-bound/local search algorithm to find the most stable schedule among schedules with minimal maximum tardiness. And Hoogeveen; Lenté and T'kindt (2012) studied new job insertion problem in single machine schedule to minimize total cost of disruption and to minimize number of setups.

Unal, Uzsoy and Kiran (1997) considered the problem of rescheduling a facility modeled as a single machine in the face of newly arrived jobs with part-type dependent setup times. They provided a polynomial time algorithm to minimize the maximum completion time of the new jobs.

Yang (2007) considered single machine rescheduling problem for new jobs arrival, in which jobs processing time can be reduced at a cost, with the objective to

minimize total cost after rescheduling. (LIU; ZHOU, 2015) studied the rescheduling problem of a single machine in response to an unexpected arrival of new jobs, where both the deterioration and learning effects are considered.

Zakaria and Petrovic (2012), Moratori and Petrovic (2012), Vallada and Ruiz (2010), Kundakci and Kulak (2016), Armentano and Mazzini (2000) and Ribeiro and Souza (2009) have used the Genetic Algorithms to solve different scheduling problems. They have reported that GA can achieve good solutions in a time efficient way.

As seen from the literature review, most of authors have adopted the reactive strategy for rescheduling, in which an initial schedule is generated then it's updated according to a specific policy to minimize its impact on system performance. Also, approximate methods for optimization have been used widely since most of scheduling problem are NP-hard.

Quite recently, researchers started to use stability based performance measures for the rescheduling problem to reduce systems nervousness. Most researches on single machine rescheduling problem have considered the disruption of new jobs arrival because of the fact that manufacturing systems are dynamic and new jobs continue to arrive while executing the initial schedule.

But the disruption of machine breakdown has been merely considered in single machine rescheduling problem despite of its importance. Also, most of the previous studies did not take into account the effect of disruption timing although it may affect the rescheduling decision.

In this paper, the single-machine rescheduling problem with unexpected disruptions is addressed. A genetic algorithm is used to generate both initial schedule and the updated one according to a reactive strategy. This study compares the performance of three rescheduling policies; right-shift rescheduling, event-driven rescheduling, and periodic rescheduling policy. Total tardiness is used as a measure of schedule efficiency. And total cost is used as a combined efficiency and stability measure. And there has been no reported work on this issue so far.

The rest of the paper is organized as follows: In Section 2, the proposed study will be detailed. Section 3 is devoted to computational results and discussion. The conclusion is reported in Section 4.



2. THE PROPOSED STUDY

This section describes the scheduling/rescheduling problem, the experimental design, and the proposed model under which the rescheduling policies are studied.

2.1. Problem description

This work considers the problem of scheduling/rescheduling of a single machine under two different objectives; minimizing the total tardiness and minimizing the total cost. At the beginning of the scheduling horizon there is a certain set of jobs to be scheduled. So, it is a static scheduling problem. Different rescheduling approaches are adopted and the performances of different rescheduling policies are to be compared in terms of total tardiness and total cost.

This study is based on the following assumptions:

- The planning horizon is a single period (one month) and jobs that arrive during schedule execution will be delayed till the end of current schedule (Frozen MPS).
- The processing times and due dates for jobs are deterministic and dependent on the jobs.
- All jobs are available at time zero.
- Every single job is assumed to be a batch which can't be split.
- Processing of disrupted jobs is to be resumed immediately after disruption is over.
- Machine setup time is included in the processing time for each job and it is independent of the preceding job.
- The time of disruption occurrence and the duration of downtime resulted from each disruption are probabilistic, following a uniform probability distribution.
- The machine can process only one job at a time.
- Materials are delivered to the shop floor according the initial schedule.

- The required raw material for each job is to be supplied just before the starting time of the job.
- WIP inventory is charged with holding cost according to the time spent in manufacturing workshop.

During the execution of the schedule, unexpected disruptions may occur. To overcome the forthcoming of these disruptions, the following rescheduling actions may be applied as policies:

- a) **Right-shift rescheduling:** Unprocessed jobs' start times are shifted forward by the amount of disruption durations keeping the initial jobs' processing sequence.
- b) **Event-driven rescheduling:** complete rescheduling of all unprocessed jobs at time of disruption.
- c) **Periodic rescheduling:** rescheduling is made periodically with fixed intervals of time. If disruptions took place before the beginning of any interval, the Right-Shift rescheduling action is taken until the processing situation is revised at the beginning of each interval for rescheduling.

The following are the problem parameters considered in this study are:

- a) **Rescheduling policy:** This refers to whether schedule will be revised periodically at equal time intervals, or only at the time of disruption or will not be revised at all.
- b) **Time of disruption occurrence:** This refers to the time of disruption occurrence relative to the planning period.
- c) **Number of disruption occurrences:** this refers to the number of disruptions that occur during the planning period.
- d) **Disruption duration:** This refers to the duration of the downtime due to disruption.
- e) **Jobs' processing time (p_j):** this refers to the jobs' processing time compared to the disruption duration. It is expressed in terms of the number of jobs to be scheduled at the same planning horizon. Such that smaller number of jobs means longer processing times and the vice versa.

Jobs' processing times are N uniformly distributed random variables such that $p_j \in U[6, 24]$, $j = 1, \dots, N$. Where N is the number of jobs.

- f) **Due date tightness:** this indicates how jobs' due dates are spaced in time. The following tightness measure suggested by Pinedo (2010) is used in this study:

$$\varphi = 1 - \frac{\sum d_j}{N C_{max}} \quad (1)$$

Where $\frac{\sum d_j}{n}$ is the average of the due dates, and C_{max} is the makespan. Values of φ close to 1 indicate that the due dates are tight and values close to 0 indicate that the due dates are loose.

Jobs' due dates, d_j , are N uniformly distributed random variables such that $d_j \in U[8, 176*\beta]$. Where β is a percentage factor controlling due date tightness and it will be initially set to 1.

2.2. Mathematical Modelling

The mathematical modelling of the problem is based upon the work of Armentano and Mazzini (2000) on a similar problem with some modifications to adapt the assumptions of this study.

Also the rescheduling cost function used in this study is based upon the cost function proposed by Naseri and Kuzgunkaya (2010) for a flexible manufacturing system (FMS) with some modifications to adapt single machine rescheduling costs.

Notations, sets, and variables used in this model are presented in Table 1.

Table 1: Notations, sets and variables

N	Total number of jobs to be scheduled
J	The job number, where $j = 1, \dots, N$.
R	Rescheduling frequency (total number of rescheduling times).
R	The number of the rescheduling time, where $r = 1, \dots, R$.
D	Total number of disruptions.
D	The disruption number, where $d = 1, \dots, D$.
Y	The initial schedule.
φ	Due date tightness factor, where $0 < \varphi < 1$.
Q_r	Time of the rescheduling (r) [in hrs.].
J_0	Set of jobs to be initially scheduled, where $J_0 = \{1, \dots, N\}$.
T_j	Tardiness of job (j) [in hrs.].
P_j	Processing time of job (j) [in hrs.].
d_j	Due date of job (j) [in hrs.].
C_j	Completion time of job (j) [in hrs.].
S_j	Starting time of job (j) [in hrs.].

S_j^{new}	Starting time of job (j) in the updated schedule [in hrs.].
$S_j^{initial}$	Starting time of job (j) in the initial schedule [in hrs.].
U_c^*	Partial schedule of the completed jobs.
U_{uc}^*	Partial schedule of the uncompleted jobs.
U^*	Actually realized schedule.
$S_{kj}(u)$	Start time of jobs k in schedule u [in hrs.].
$C_{kj}(u)$	Completion time of job k in schedule u [in hrs.].
π^*	Optimized jobs' sequence.
DS_d	Start time of disruption (d) [in hrs.].
DE_d	End time of disruption (d) [in hrs.].
DL_d	Duration of disruption (d) [in hrs.].
Y_{jd}	Binary variable that indicates whether the disruption (d) occurred during the processing of job (j).
A	Tardiness penalty cost for each time unit of tardiness [in L.E./hr.].
M	WIP holding cost for each time unit of holding [in L.E./hr.].
T	Material expediting cost for each time unit of expediting [in L.E./hr.].
Y	Scheduling / rescheduling cost for each schedule generation [in L.E./Schedule].

The objective functions are:

$$\text{Min. } Z = \sum_{j=1}^N T_j \quad (3)$$

$$\text{Min. } Z = C_{\text{tardiness}} + C_{\text{WIP}} + C_{\text{expediting}} + C_{\text{sch/res}} + C_{\text{earliness}} \quad (4)$$

Subject to:

$$S_{j+1} = S_j + P_j + Y_{jd}DL_d \quad j = 1, 2, \dots, k-1; \text{ where } k \text{ is disrupted job}; \forall d \quad (5)$$

$$\forall d \quad (6)$$

Where:

$$DS_d \leq C_k \leq DE_d \quad d = 1, 2, \dots, D \quad (7)$$

$$T_j = \max(0, C_j - d_j) \quad \forall j \quad (8)$$

$$0 < DS_d < C_N \quad \forall d \quad (9)$$

$$Q_r = \begin{cases} DS_d & \text{For Event - driven policy,} \\ \frac{\sum_{j=1}^N P_j}{R+1} & \text{For Periodic Policy,} \end{cases} \quad \forall d, \quad l = 1, \dots, R \quad (10)$$

$$C_{\max}(\pi^*) = \sum_{j=1}^N P_j \quad (11)$$

$$C_{\text{tardiness}} = \sum_{j=1}^N T_j \alpha \quad (12)$$

$$C_{\text{WIP}} = \sum_{\sigma=1}^Z (S_{\sigma}^{new} - S_{\sigma}^{initial}) \mu \quad S_i^{new} > S_i^{initial} \quad (13)$$

$$C_{expediting} = \sum_{j=1}^N \max(S_j^{initial} - S_j^{new}, 0) \tau \quad S_j^{initial} > S_j^{new} \quad (14)$$

$$C_{earliness} = \max(C_j - d_j, 0) \mu \quad \forall j \quad (15)$$

$$C_{sch/resz} = R * Y \quad \forall j \quad (16)$$

$$C_j, P_j, d_j, DS_d, DE_d, T_j \geq 0 \quad \forall j, \forall d \quad (17)$$

Eq. (3) represents the objective of minimizing the total tardiness, which is the summation of tardiness for all jobs, T_j . Eq. (4) represents the other objective of minimizing the total cost, which is the summation of the costs of tardiness ($C_{tardiness}$), earliness ($C_{earliness}$), material expediting ($C_{expediting}$), WIP holding (C_{WIP}) and scheduling/rescheduling.

Expression (5) implies that the start time of a particular job (S_{j+1}) is greater than or equal to the summation of the processing time of the previous job (P_j), previous jobs start time (S_j) and the disruption duration DL_d , if any exists then ($Y_{jd} = 1$) otherwise ($Y_{jd} = 0$).

Expression (6) shows that Jobs can't be split such that after disruption is over, the disrupted job starts processing from the same point where it was before disruption. So the completion time of the disrupted job (k) is equal to the sum of completion time of previous job (C_{k-1}), processing time of the disrupted job (P_k) and the down time due to disruption (DL_d).

Expression (7) defines the disrupted job (k) whose completion time, C_k , lies between disruption, D_d , start time (DS_d) and disruption end time (DE_d).

Expression (8) defines job tardiness, T_j , as the positive difference between job completion time, C_j , and its due date, d_j . Expression (9) implies that all disruptions occur within schedule makespan.

Expression (10) defines the rescheduling start time, Q_r , according to the rescheduling policy. Such that it equals the disruption start time, DS_d , for event-driven policy. And it equals the rescheduling period multiplied by an integer whose value changes from 1 to the rescheduling frequency, R .

Eq. (11) implies that the initial schedule has no idle time between jobs, and the starting time of the first job is zero.

Eq. (12) represents the penalty cost of jobs' tardiness. Eq. (13) represents the cost of holding material for jobs which were the next in order when the rescheduling triggered and their start time changed in the new schedule, S_z^{new} , to start later than the initial schedule, $S_z^{initial}$. Since their material is already delivered to the shop floor.

Eq. (14) represents the cost of expediting material for jobs whose start time has changed in the new schedule to be started earlier than the initial schedule. Eq. (15) represents the cost of earliness. Eq. (16) represents the cost of scheduling/rescheduling which is the cost of schedule computation and the wage of the scheduler. Expression (17) shows that Job processing time, job due date, job completion time, disruption start time, disruption end time, and job tardiness are all non-negative values.

2.3. The genetic algorithm (GA)

In the used GA, the chromosome encoding is a job-based representation. Each chromosome represents a processing sequence. An inversion mutation operator is used in which two random positions in the parent are chosen and the genes between the two points are inverted, while those outside the points are retained.

Two crossover operators are used simultaneously in the GA, namely *precedence preservative* and *set-partition* operators. One child is produced from two parents once by the Set-Partition crossover operator, and another child is produced from the same parents by the Precedence Preservative crossover operator. Both children are then evaluated according to the fitness function, and the child that has the better fitness value proceeds to the following generation.

The GA parameters that are used are presented in Table 2.

Table 2: Genetic algorithm parameters

Parameter	Value
Population size	50
Cross over rate	0.9
Mutation rate	0.1
No. of elite children	1
No. of generations	1000

The cross over rate of 0.9 and the number of elite children of 1 are determined through a factorial experiment (ANOVA technique) reported by Church and Uzsoy (1992) for the same cross over and mutation operators.

2.4. Experiments design

Production rescheduling may be affected by many factors and our concern is to estimate the effect of these factors on the rescheduling decision, as well as the effect of possible interactions between these factors. In each complete trial of the experiment all possible combinations of all levels of the factors are investigated.

Using Minitab® statistical software, a general factorial experiment, in which more than two factors are considered and each factor has different number of levels, is designed to study the performance of different rescheduling policies in terms of the total tardiness and the total cost as performance measures.

The experiments include the following factors: 1) rescheduling policy. 2) time of disruption occurrence. 3) Number of disruptions during the planning period. 4) Disruption duration. 5) jobs processing time represented by different number of jobs to be scheduled in the same planning horizon. 6) the optimization objective (minimum total tardiness or minimum total cost). The experiment is designed to consider different disruption parameters to optimize a certain objective of interest. Table 3 shows the designed experiments for this study.

The disruption parameters are the number of disruption occurrences, disruption duration and the time of disruption occurrence.

The number of disruptions, D , is an input variable and it changes to reflect different levels of disruption frequency (Table 3). The disruption duration, DL_d , is an input variable and it changes to reflect different levels of the downtime resulted from disruptions. According to the number of disruptions the model generates random uniformly distributed values, DL_d , based on the level of disruption duration (Table 3).

The time of disruption occurrence, DS_d , is an input variable representing disruption start time and it changes to reflect the fact that the disruption can start at different times of schedule execution. Such that it can start early, at the middle or late in the makespan. The model generates random uniformly distributed values, DS_d , within a predefined time interval based on the level of disruption time (Table 3).

Table 3: Experiments design

	Experimental factor					
	Policy	Time of disruption	Number of disruptions	Disruption duration	Jobs' Processing time	Optimization Objective
Factor levels	Event-driven	Early [0.05M**, 0.35M]	None (0)	Short [0.02 TWK***, 0.04 TWK]	Small (10 jobs)	Min. total tardiness
	Periodic (r*=0)		Small (1)			
	Periodic (r=2)	Middle [0.35M, 0.65M]		Medium (2)	Long [0.07 TWK, 0.14 TWK]	Large (25 jobs)
	Periodic (r=4)					
	Periodic (r=6)	Late [0.65M, 0.95M]	Large (3)			
	Periodic (r=8)					
	Periodic (r=10)					
	Periodic (r=20)					

* r is the rescheduling frequency.
 ** M is the makespan of the initial schedule.
 *** TWK is the total work content ($\sum_{j=1}^N P_j$).

2.5. Scheduling/rescheduling model

The developed production rescheduling model aims at introducing the unexpected disruption to the initial production schedule and generates an updated production schedule. The rescheduling model is composed of two stages; the first one is for generating the initial schedule and the second is for updating the initial schedule according to the policy followed.

2.6. Generating initial schedule

Let $J_0 = \{1, \dots, N\}$ be the set of original jobs that are scheduled for processing on the machine. The model uses this set of jobs to construct a row vector which is used as the chromosome for the genetic algorithm. The genetic algorithm finds the best sequence, π^* , that optimizes the objective function.

So, it is assumed that the jobs of π^* have been sequenced optimally to minimize the objective function at time zero. Then, the model constructs an initial schedule, u , for the jobs sequence π^* . The decision variable is jobs' start times in schedule u , $S_{[j]}(u)$, for $j \in J_0$. Let $C_{[j]}(u)$ be the completion time of job j in schedule u and $\pi_{[k]}^*$ is the job in the k^{th} position in the sequence π^* .

Algorithm 1: Constructing initial schedule

Step 1: Use set of original jobs, J_o , to form a row vector x .

Step 2: Set schedule start time equal zero.

Step 3: Input vector x to GA to get optimized sequence π^* .

Step 4: Construct initial schedule, u , according to π^* .

2.7. Generating updated schedules

The model uses one of three different algorithms to generate the updated schedule according to the rescheduling policy followed. In the following, there is a set of disruptions start time, $DS_d = \{DS_1, \dots, DS_D\}$, which are randomly generated.

Algorithm 2: Right-shift rescheduling

According to the right-shifting policy, the algorithm keeps the initial sequence of jobs. And only right-shifts the start time of all jobs that start after the disruption by the amount of disruption duration, DL_d .

Step 1: Generate disruption parameters (No. of disruptions, timing, and duration).

Step 2: For each disruption, D_d , find the disrupted job k where $S_{[k]}(u) \leq DS_d \leq C_{[k]}(u)$.

Step 3: Identify the partial schedule of completed jobs, $u_c^* = J_o\{1, \dots, k\}$.

Step 4: Add disruption duration to the completion time of job k in u_c^* .

Step 5: Identify another partial schedule of the unprocessed jobs, $u_{uc}^* = J_o\{k+1, \dots, N\}$.

Step 6: Add disruption length, DL_d , to start and completion time of all jobs in u_{uc}^* .

Step 7: Merge the partial schedules, u_c^* and u_{uc}^* into u^* .

Step 8: set $u = u^*$.

Step 9: Repeat step 2 through 8 for all disruptions D .

Step 10: Calculate total tardiness and rescheduling total cost on realized schedule, u .

Algorithm 3: Event-driven policy to update schedule

In event-driven rescheduling policy the model generates a new schedule, u^* , for the unprocessed jobs, J_{uc} , using the GA at the time of disruption, DS_d .

Step 1: Generate disruption parameters (No. of disruptions, timing, and duration).

Step 2: For each disruption, D_d , find the disrupted job k where $S_{[k]}(u) \leq DS_d \leq C_{[k]}(u)$.

Step 3: Identify partial schedule of completed jobs, $v_c^* = J_o\{1, \dots, k\}$.

Step 4: Add disruption duration to completion time of jobs k in v_c^* .

Step 5: Identify partial schedule of unprocessed jobs, $J_{uc} = J_o\{k+1, \dots, N\}$.

Step 6: Set start time of partial schedule of unprocessed jobs, v_{uc}^* , equal $C_{[k]}(u) + DL_d$.

Step 7: Run GA for all jobs in J_{uc} and get updated jobs sequence π^* .

Step 8: Construct updated partial schedule of unprocessed jobs, v_{uc}^* , according to updated sequence π^* .

Step 9: Merge partial schedules, v_c^* and v_{uc}^* into v^* .

Step 10: set $u = v^*$.

Step 11: Repeat step 2 through 10 for all disruptions D .

Step 12: Calculate total tardiness and rescheduling total cost on realized schedule, u .

Algorithm 4: Periodic policy to update schedule

In periodic rescheduling policy, in addition to the set of disruptions, there is a set of predefined rescheduling times, $Q_r = \{Q_1, \dots, Q_R\}$, equally spaced along the planning horizon based on the given rescheduling frequency. At each rescheduling time, Q_r , the model generates a new schedule for the unprocessed jobs.

Step 1: Generate disruption parameters (No. of disruptions, timing, duration).

Step 2: For each disruption, D_d , find the disrupted job k where $S_{[k]}(u) \leq DS_d \leq C_{[k]}(u)$.

Step 3: Identify partial schedule of completed jobs, $v_c^* = J_o\{1, \dots, k\}$.

Step 4: Add disruption duration to completion time of jobs k in v_c^* .

Step 5: Identify partial schedule of unprocessed jobs, $v_{uc}^* = J_o\{k+1, \dots, N\}$.

Step 6: Add disruption length, DL_d , to start and completion time of all jobs in v_{uc}^* .

Step 7: Merge partial schedules, v_c^* and v_{uc}^* into v^* .

Step 8: set $u = v^*$.

Step 9: For each rescheduling time, Q_r , find job Y where $S_{[Y]}(u) \leq Q_r \leq C_{[Y]}(u)$.

Step 10: Identify partial schedule of processed jobs, $J_{uc} = J_o\{Y+1, \dots, N\}$.

Step 11: Run GA for all jobs in J_{uc} and get updated jobs sequence π^* .

Step 12: Construct updated partial schedule of unprocessed jobs, v_{uc}^* , according to updated sequence π^* .

Step 13: Merge partial schedules, v_c^* and v_{uc}^* into v^* .

Step 14: set $u = v^*$.

Step 15: Repeat step 2 through 14 for all rescheduling times Q_r .

Step 16: Calculate total tardiness and rescheduling total cost on realized schedule, u .

3. RESULTS AND DISCUSSION

This section provides the computational results of different rescheduling policies at different disruption parameters. Results are classified in sections according to the performance criteria.

3.1. Minimum total tardiness

The three rescheduling policies under examination are: right-shift rescheduling policy, event driven policy and periodic policy. Each policy is tested under different disruption parameters. The disruption parameters are the number of disruptions, disruption duration and the time of disruption occurrence. The objective function is to minimize the total tardiness.

Figure 1 shows the performance of three rescheduling policies in terms of total tardiness for different disruption parameters. Each value in the graphs is an average of 10 trials.

Figure 1 shows the total tardiness at six different disruption parameters for 25 jobs problem. The left and right panels show the results for short and long disruption duration, respectively. From the graph it is seen that as the disruption starts late, the total tardiness decreases, and performance of the three rescheduling policies are very close.

This may be due to the fact that the later the disruption begins the less the number of affected jobs. Hence the smaller total tardiness. Also when scheduling/rescheduling has the objective of minimizing total tardiness, jobs with largest slack time come at the end of the schedule. So, the effect of the disruption can be absorbed due to slack time.

Also, it can be noted that the earlier the disruption begins the more the affected number of jobs and the more the jobs tend to miss its due date, hence the total tardiness increases significantly. The periodic policy tends to be superior to the other rescheduling policies for long disruptions while the event-driven policy tends to be superior to other policies for short disruptions.

Figure 2 shows the results of rescheduling policies under the same disruption parameters (long, middle disruption) for different problem sizes. The graph shows that when the machine is prone to long disruption duration, the periodic policy will result in the minimum total tardiness regardless of the problem size.

This is because some jobs may have tight due dates, so when generating the initial schedule, schedules that don't begin with these jobs will be excluded because of large total tardiness. But after the completion of these jobs, there's a better opportunity for the scheduling algorithm to find another schedule that will result in lower total tardiness than that which resulted from keeping the initial schedule.

The opportunity for finding better solutions arise from the fact that the tardiness of jobs whose completion time is less than its due date is zero, so the starting time of these (early) jobs can be right-shifted to some extent, $|d_j - C_j|$, without increasing its tardiness. Hence, the tardy jobs can be rescheduled to start earlier so its tardiness decreases, and the total tardiness decreases.

Figure 3 is the same as Figure 2 but for short middle disruption. The graph shows that event driven policy outperforms other rescheduling policies in terms of total tardiness when the machine is prone to short disruptions. This is because rescheduling at time of short disruption makes benefit of the available slack time of greater number of jobs than that number of jobs available when waiting until the next rescheduling time in the periodic policy.

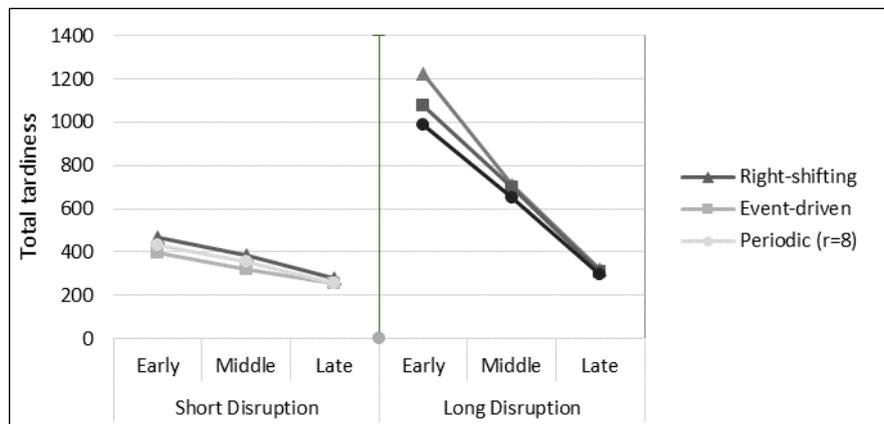


Figure 1: Total tardiness resulted from different rescheduling policies at different disruption durations and different time of disruption occurrence and problem size 25 job.

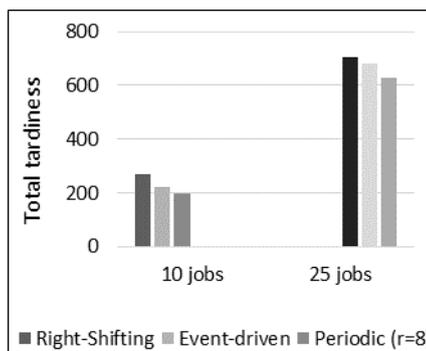


Figure 2: Total tardiness for different rescheduling policies with long middle disruption and different problem size.

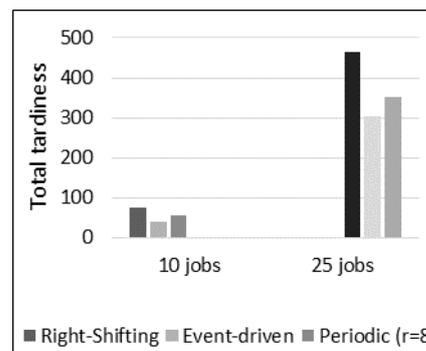


Figure 3: Total tardiness for different rescheduling policies with short middle disruptions and different problem sizes.

It can be noted from Figure 2 and Figure 3 that the total tardiness increases with increasing the number of jobs. The reason is that the due dates of the larger problem is more tight than that of the small one, because both sets of due dates have been generated from the same uniform distribution using the same parameters $U[8, 176]$. By calculating the tightness measure, ϕ , for both sets of due dates it will result in 0.33 and 0.68 for the small and large problem, respectively. So the jobs with more tight due dates are more likely to miss their due dates. Hence they have higher total tardiness.

The effect of the rescheduling frequency on the total tardiness is shown in Figure 4. The graph shows the percentage improvement in total tardiness for periodic rescheduling with different frequency at different values for the total disruption duration compared to the right-shifting policy.

From the graph it can be noticed that at low-to-moderate values of total disruption time increasing the rescheduling frequency may reduce the total tardiness (higher percentage improvement). This may be due to the utilization of the slack time for some jobs to decrease the tardiness of other jobs explained earlier.

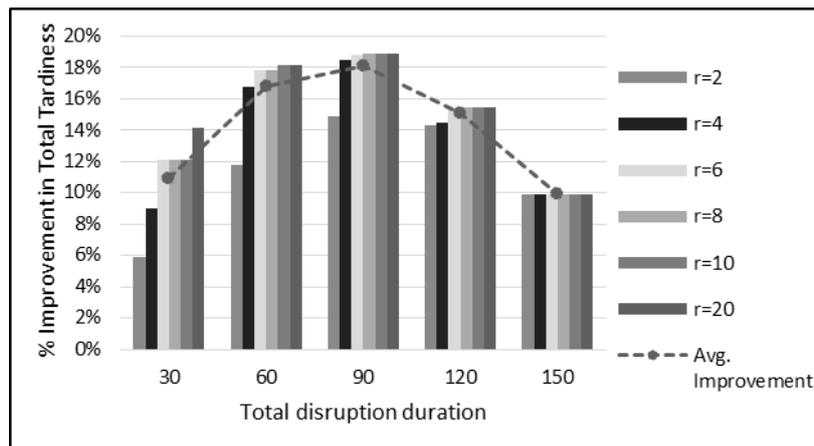


Figure 4: Percentage improvement in total tardiness due rescheduling for different rescheduling frequency at different values for total disruption time compared to right-shifting policy.

Also it can be noted from Figure 4 that the average improvement (*dashed-line*) compared to the right-shifting policy increases as the total disruption duration increases to a some value of the total disruption duration after which the improvement that can be obtained from rescheduling decreases regardless of the rescheduling frequency.

For example, at total disruption duration of 150 the rescheduling frequency of 2 will yield the same improvement as rescheduling frequency of 20. This behavior may be due to the fact that at some value of the total disruption time, the due date of the farthest job will be equal to the summation of remaining jobs processing time and total disruption time as follows:

$$\max(d_j) = \sum_{j=k+1}^N p_j + \sum_{i=1}^D DL_i \quad (18)$$

This means that, at this value the slack time of all remaining jobs is less than or equal zero (no slack time). Hence, no more right-shifting is possible without increasing the total tardiness.

The effect of the number of disruptions on the performance of different rescheduling policies is shown in Figure 5. The figure shows the total tardiness resulted from rescheduling with different frequencies at different number of disruptions with the same total disruption duration.

It can be concluded from the figure that frequent short disruptions results in smaller total tardiness than that results from one long disruption. This is because long disruptions at the beginning of schedule affect larger number of jobs than that affected by short successive disruptions (Figure 1). On the other hand, the effect of small disruptions can be reduced by frequent rescheduling utilizing the slack time of the remaining jobs.

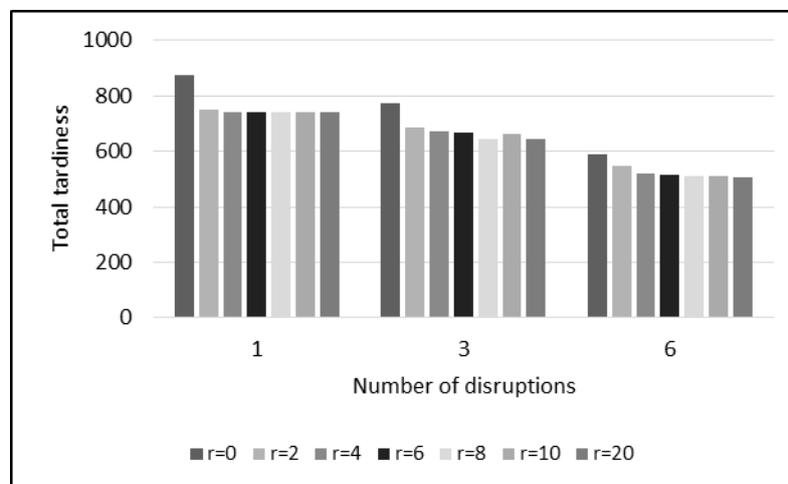


Figure 5: Total tardiness resulted from different rescheduling frequencies at different number of disruptions with the same total disruption duration

3.2. Minimum total cost

In this subsection the same rescheduling policies are examined but the objective function is to minimize the total cost. Figure 6 shows the total cost resulted from different rescheduling policies at different disruption duration and time of disruption occurrence. It can be noticed from the graph that the total cost is higher for early disruptions. This is because of the increased total tardiness resulted from early disruptions so the tardiness penalty cost increases significantly (Figure 1).

Also it can be noticed from Figure 6 that right-shifting policy will result in the minimum total cost. This is because of the high cost of scheduling/rescheduling

relative to other cost elements. Although right-shifting policy will result in higher tardiness than other rescheduling policies (Figure 1), but the increased tardiness cost will be less than the rescheduling cost incurred due to increased rescheduling frequency.

The results show that the ratio between the scheduling/rescheduling cost and the tardiness cost, Υ / α , has a significant effect on the performance of rescheduling policies in terms of the total cost. Figure 7 Shows the total rescheduling cost for different rescheduling frequencies at different values of the ratio between the scheduling/rescheduling cost and the tardiness penalty cost.

Results represented in Figure 7 are at the same number of disruptions and total disruption duration. It can be noticed from Figure 7 that at relatively small ratios of scheduling cost to the tardiness penalty, high rescheduling frequency results in reduction of the total costs due to the corresponding reduction of total tardiness (Figure 4). But as this ratio increases the rescheduling becomes too expensive and keeping the initial schedule results in minimum total cost despite the increased total tardiness.

Figure 8 and Figure 9 show the total cost results for different rescheduling policies for different problem sizes at middle short and long disruptions, respectively. It can be noticed from the graph, the total cost increases significantly when problem size increases. This is because of the increased tardiness that resulted from increased tightness for larger problem size which was explained in section 3.1.

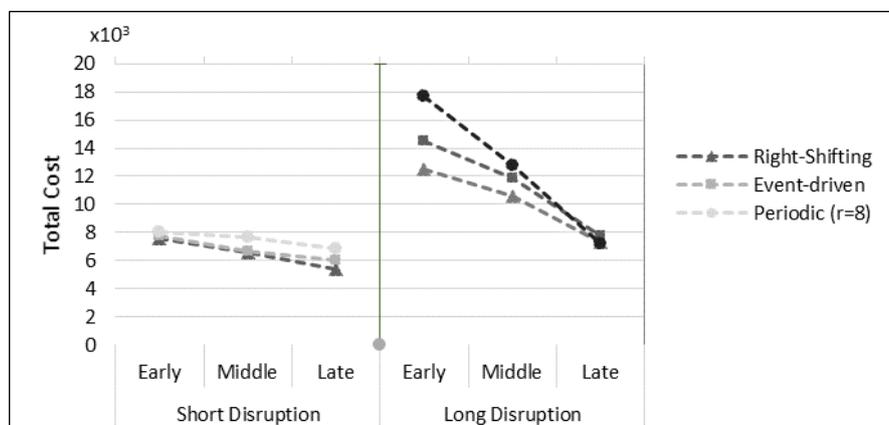


Figure 6: Total cost resulted from different rescheduling policies at different disruption durations and different time of disruption occurrence and problem size 25 job.

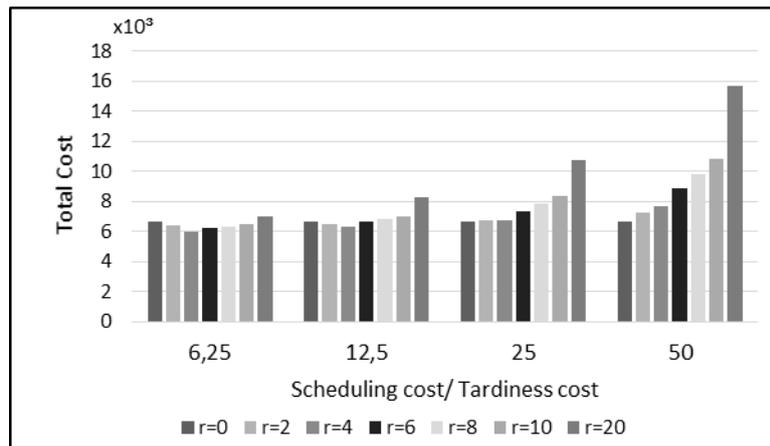


Figure 7: Total cost of rescheduling for different rescheduling frequencies at different values of scheduling/tardiness cost ratio.

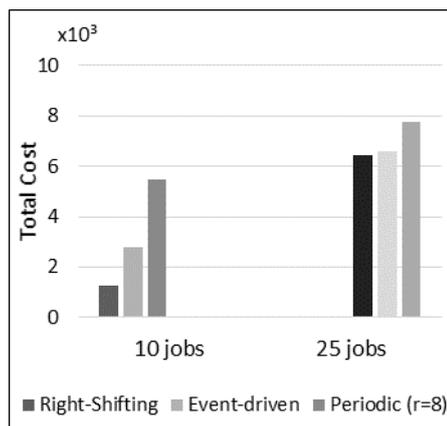


Figure 8: Total cost resulted from different rescheduling policies at short middle disruption for different problem size.

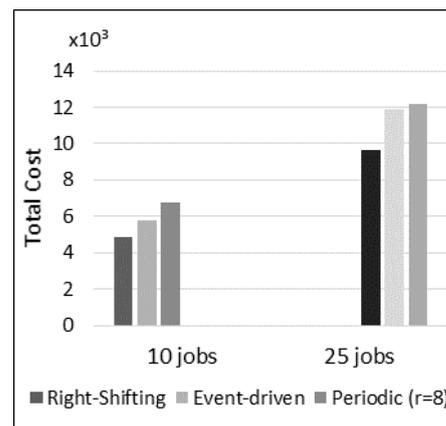


Figure 9: Total cost resulted from different rescheduling policies at long middle disruption for different problem size.

3.3. Sensitivity analysis

The main purpose of sensitivity analysis in this study is to identify the sensitive parameters, that changes in their values could result in significant changes in the rescheduling decision. In this section, the sensitivity of our results to changes in due date tightness coefficient and cost coefficients is tested.

3.4. Due date tightness

In studying the sensitivity of tardiness results to changes in due date tightness, three values of tightness factor, ϕ : 0.4, 0.6 and 0.8 are used to represent three levels of due date tightness following the work of Armentano and Mazzini (2000). The due date tightness factor is calculated from Eq. (1) and it has a maximum value of 1. Such that, values of ϕ close to 1 indicate that the due dates are tight and values close to 0 indicate that the due dates are loose.

The effects of due date tightness have been tested under the following conditions:

1. Rescheduling policy: Event driven and Periodic ($r=0, 4, 8, 20$)
2. Objective function: minimum total tardiness.
3. Problem size: large (25 jobs).
4. The effect of rescheduling is expressed in terms of the percentage improvement which is defined as the ratio between rescheduling policy's result and the right-shifting policy's result.

Figure 10 show the total tardiness that results from different rescheduling policies due to changing the due date tightness factor for a large problem size (25 jobs). The results in the graph are obtained at the disruption parameters of three long middle disruptions. It can be noticed from Figure 10, the total tardiness is sensitive to changes in due date tightness factor between the values of 0.4 and 0.8. And the total tardiness increases as the due dates tightness increases. This is because more jobs tend to miss their due date when due dates are too tight.

Also, it can be noticed from Figure 10 that, when due dates are loose, periodic rescheduling will be superior to the other policies. This is because jobs with loose due dates have more slack time. So more frequent rescheduling in periodic rescheduling policy makes better utilization of the available slack time for all jobs.

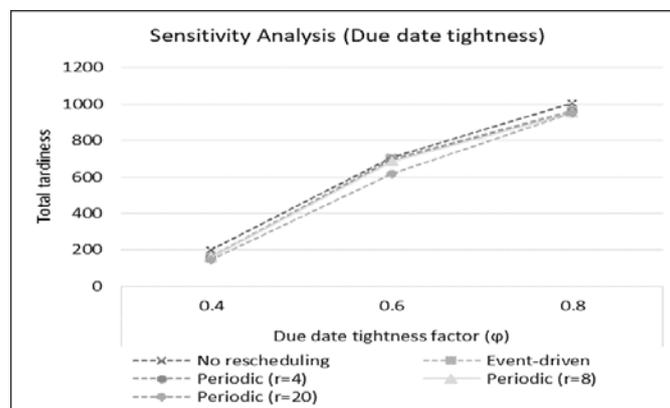


Figure 10: Total tardiness changes due to changing due date tightness factor with long middle disruption for 25 jobs problem.

In Table 4, the total tardiness results for different rescheduling policies and the percentage improvement in performance at different levels of due date tightness factor are presented. It can be noticed that, for tight due dates all rescheduling policies don't result in significant improvement in performance (Table 4) and they

performs almost the same. This is because of that, at tight due dates jobs don't have much slack time and there's less opportunity to utilize this slack time to generate better schedule with lower total tardiness. As jobs' due dates get looser, the rescheduling frequency becomes more significant and this is consistent with the results reported by Church and Uzsoy (1992).

Table 4: Effect of due date tightness and rescheduling policy on total tardiness

Policy	Tightness factor (ϕ)		
	0.4	0.6	0.8
Right-shifting ($r=0$)	198	707	1003
Event-driven (% Improvement)	163 (18%)	699 (1%)	967 (4%)
Periodic ($r=4$) (% Improvement)	159 (20%)	693 (2%)	957 (5%)
Periodic ($r=8$) (% Improvement)	157 (21%)	688 (3%)	953 (5%)
Periodic ($r=20$) (% Improvement)	145 (27%)	617 (13%)	953 (5%)

3.5. Cost coefficients

Cost coefficients used in the cost function could be changed according to company's priorities determined by the nature of its industry. These cost coefficients considered in this study are the tardiness cost, material expediting cost and WIP holding cost. The effect of changing these cost coefficients on the total cost is studied.

Through this analysis, five levels are considered for each cost coefficient: 0.1, 0.5, 1, 5, 10. One coefficient is changed at the time and the others are held constant.

Figure 11 shows the total cost for different rescheduling policies that resulted from changing the tardiness cost coefficient. From the graph it can be concluded that the total cost is sensitive to the tardiness cost coefficient, α , since changing in its value leads to significant change in the total cost.

This is because when the value of the tardiness cost coefficient increases, jobs tend to finish earlier than their due dates. Hence incurring earliness cost in addition to the increase of the tardiness cost, so the total cost increases significantly. So, the tardiness cost coefficient can be considered as dominant.

Figure 12 shows the total cost for different rescheduling policies that resulted from changing the WIP holding cost coefficient. It can be noticed from the graph that

the total cost is robust to changes in the values of WIP holding cost coefficient, μ , between the values of 0.01 and 10, and then it becomes sensitive for greater values.

Because by significantly increasing the value of WIP hold cost coefficient, the cost of delaying the start of job processing whose material has already delivered to the shop floor and start processing another job becomes too expensive. Also the earliness cost of the jobs that finish before their due date becomes too expensive, which leads to a significant increase in the total cost.

Finally, Figure 13 shows the total cost for different rescheduling policies that resulted from changing the material expediting cost coefficient. It can be noticed from the figure that, changing the value of the expediting cost coefficient between 0.01 and 100 has almost no effect on the total cost. So, the total cost results could be considered robust to changes in this cost coefficient.

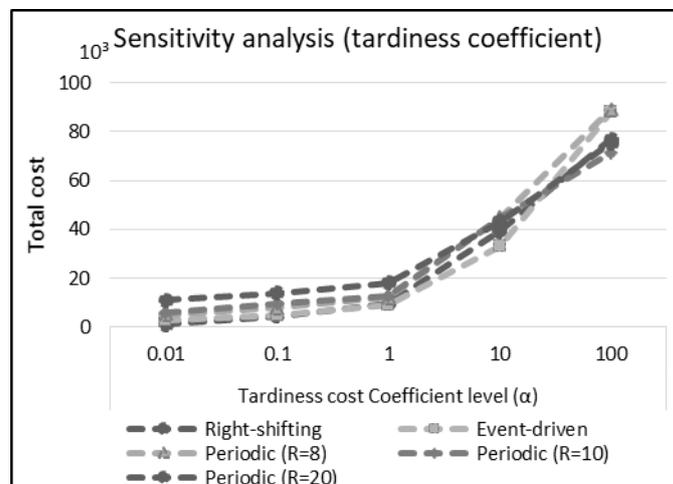


Figure 11: Effect of changing tardiness cost coefficient on total cost for different rescheduling policies at long middle disruption and 10 job problem.

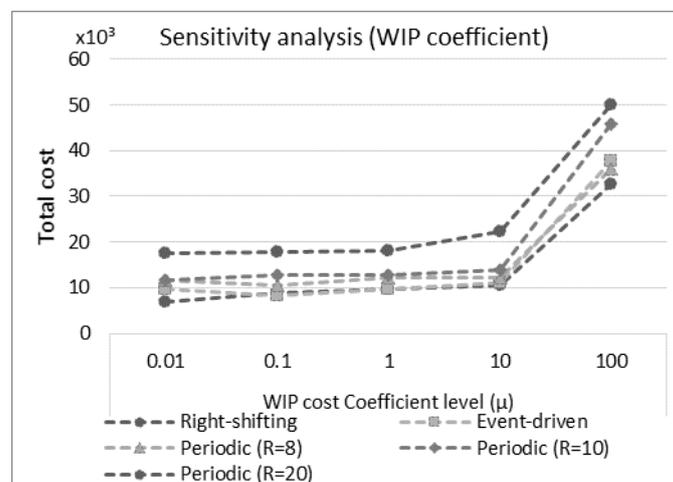


Figure 12: Effect of changing WIP holding cost coefficient on total cost for different rescheduling policies at long middle disruption and 10 job problem.

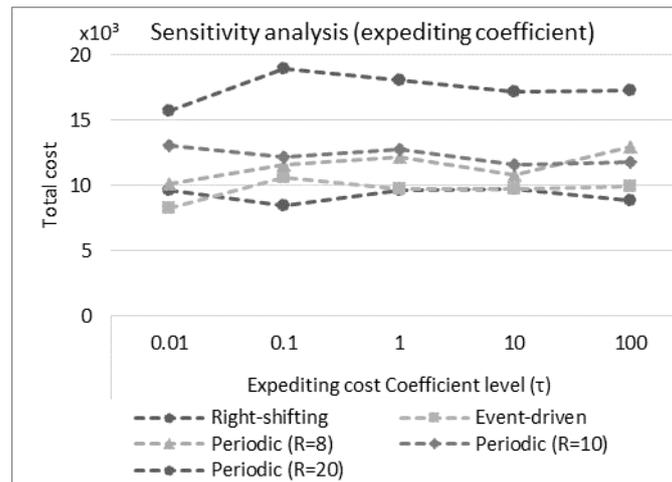


Figure 13: Effect of changing expediting cost coefficient on total cost for different rescheduling policies at long middle disruption and 10 job problem.

4. CONCLUSIONS

This study considered the rescheduling problem for a single machine with unexpected disruptions. The effect of different disruption parameters such as the time of disruption occurrence, the disruption duration, and the number of disruption occurrences was considered. The main concern of this study is to compare the performance of different rescheduling policies under different levels of disruptions' parameters to optimize a certain objective of interest. The results of our study showed that:

1) When the objective is to minimize the total tardiness:

- If the machine is prone to short disruptions, from 2% to 4% of TWK, the event-driven policy will be the superior to other policies for the early and middle disruptions regardless of jobs' processing time.
- If machine is prone to long disruptions, from 7% to 14% of total work content (TWK), the periodic policy will be superior to the other policies for early and middle disruptions regardless of the jobs' processing time.
- If disruptions occur late in schedule planning period, for instance, after the completion of 65% of initial schedule makespan, keeping the initial schedule this will always result in minimum total tardiness.

2) When the objective is to minimize total cost:

- If the ratio of the scheduling cost to the tardiness penalty cost is more than or equal to 25, keeping the initial schedule will result in the minimum total cost.
 - If the ratio of the scheduling cost to tardiness penalty cost is less than 25, the event-driven policy will be superior to other policies for short early and middle disruptions. Otherwise, the periodic policy with moderate rescheduling frequency will be superior.
- 3) As due dates become more tight, the improvement in total tardiness due to rescheduling decreases. So, event-driven policy will result in performance improvement in terms of total tardiness with much lower cost than the periodic rescheduling for jobs with tight due dates, whose tightness measure φ is greater than 0.7 .
- 4) Total cost is much more sensitive to changes in tardiness cost coefficient compared to other cost coefficients. Therefore, rescheduling becomes more efficient as the tardiness cost coefficient increases with respect to scheduling/rescheduling cost.

REFERENCES

- ABUMAIZAR, R. J.; SVESTKA, J. A. (1997) Rescheduling job shops under random disruptions, **International journal of production research**, v. 35, n. 7.
- AKKAN, C. (2015) Improving schedule stability in single machine rescheduling for new operation insertion, **Computers & Operations Research**, v. 64, p. 198-209.
- ARMENTANO, V. A.; MAZZINI, R. (2000) A genetic algorithm for scheduling on a single machine with set-up times and due dates, **Production Planning & Control**, v. 11, n. 7, p. 713-720.
- ARNAOUT, J. P. (2014) Rescheduling of parallel machines with stochastic processing and setup times, **Journal of Manufacturing Systems**.
- CARICATO, P.; GRIECO, A. (2008) An online approach to dynamic rescheduling for production planning applications, **International Journal of Production Research**, v. 46, n. 16, p. 4597-4617.
- CHIU, Y.; SHIH, C. J. (2012) Rescheduling strategies for integrating rush orders with preventive maintenance in a two-machine flow shop, **International Journal of Production Research**, v. 50, n. 20, p. 5783-5794.
- CHURCH, L.; UZSOY, R. (1992) Analysis of periodic and event-driven rescheduling policies in dynamic shops, **International Journal of Computer Integrated Manufacturing**, v. 5, n. 3, p. 153-163.



DONG, Y. H.; JANG, J. (2012) Production rescheduling for machine breakdown at a job shop, **International Journal of Production Research**, v. 50, n. 10, p. 2681-2691.

GUO, Y.; HUANG, M.; WANG, Q.; LEON, V. J. (2016) Single-machine rework rescheduling to minimize maximum waiting-times with fixed sequence of jobs and ready times, **Computers & Industrial Engineering**, v. 91, p. 262-273.

HOOGEVEEN, H.; LENTÉ, C.; T'KINDT, V. (2012) Rescheduling for new orders on a single machine with setup times, **European Journal of Operational**, v. 223, p. 40-46.

JAIN, S.; FOLEY, W. J. (2016) Dispatching strategies for managing uncertainties in automated manufacturing systems, **European Journal of Operational Research**, v. 248, p. 328-341.

KATRAGJINI, K.; VALLADA, E.; RUIZ, R. (2013) Flow shop rescheduling under different types of disruption, **International Journal of Production**, v. 51, n. 3, p. 780-797.

KUNDAKCI, N.; KULAK, O. (2016) Hybrid genetic algorithms for minimizing makespan in dynamic job shop scheduling problem, **Computers & Industrial Engineering**, v. 96, p. 31-51.

LIU, L.; ZHOU, H. (2012) Applying Variable Neighborhood Search to the Single-machine Maximum Lateness Rescheduling Problem, **Electronic Notes in Discrete Mathematics**, v. 39, p. 107-114.

LIU, L.; ZHOU, H. (2013) On the identical parallel-machine rescheduling with job rework disruption, **Computers & Industrial Engineering**.

LIU, L.; ZHOU, H. (2015) Single-machine rescheduling with deterioration and learning effects against the maximum sequence disruption, **International Journal of Systems Science**, V. 46, N. 14, p. 2640-2658.

MASON, S. J.; JIN, S.; WESSELS, C. M. (2004) Rescheduling strategies for minimizing total weighted tardiness in complex job shops, **International Journal of Production**, v. 42, n. 3, p. 613-628.

MORATORI, P.; PETROVIC, S. (2012) Match-up approaches to a dynamic rescheduling problem, **International Journal of Production Research**, v. 50, n. 1, p. 261-276.

NASERI, E.; KUZGUNKAYA, O. (2010) Cost effective handling of disruptions in production management, in **POMS 21st Annual Conference**, Vancouver.

PINEDO, M. L. (2010) **Scheduling: theory, algorithms and systems**, New York: Springer, p. 379.

RIBEIRO, F. F.; SOUZA, S. R. (2009) An Adaptive Genetic Algorithm for solving the Single Machine Scheduling Problem with Earliness and Tardiness Penalties, **International Conference on Systems, Man, and Cybernetics**.

UNAL, A. T.; UZSOY, R.; KIRAN, A. S. (1997) Rescheduling on a single machine with part-type dependent setup times and deadlines, **Annals of Operations Research**, v. 70, p. 93-113.

VALLADA, E.; RUIZ, R. (2010) Genetic algorithms with path relinking for the minimum tardiness permutation flowshop problem, **Omega**, v. 38, p. 57-67.



VIEIRA, G. E.; HERMANN, J. W.; LIN, E. (2000) Predicting the Performance of Rescheduling Strategies for Parallel Machine Systems, **Journal of Manufacturing Systems**, V. 19, n. 4.

VIEIRA, G. E.; HERMANN, J. W.; LIN, E. (2000) Analytical models to predict the performance of a single-machine system under periodic and event-driven rescheduling strategies, **International journal of production**, v. 38, n. 8, p. 1899 - 1915.

VIEIRA, G. E.; HERMANN, J. W.; LIN, E. (2003) Rescheduling manufacturing systems: a framework of strategies, policies, and methods, **Journal of Scheduling**.

YANG, B. (2007) Single machine rescheduling with new jobs arrivals and processing time compression, **The International Journal of Advanced Manufacturing**, v. 34, n. 7, p. 378-384.

ZAKARIA, Z.; PETROVIC, S. (2012) Genetic algorithms for match-up rescheduling of the flexible manufacturing systems, **Computers & Industrial Engineering**, v. 62, p. 670-686.

