IEMS Vol. 4, No. 1, pp. 1-22, June 2005.

Heuristics for Job Shop Scheduling Problems with Progressive Weighted Tardiness Penalties and Inter-machine Overlapping Sequence-dependent Setup Times

Chatpon Mongkalig†
Industrial Engineering and Management Program, School of Advanced Technologies
328 Pichai Road, Dusit, Bangkok 10300 Thailand
Tel: +66-9-079-6000, +66-9-128-0001, E-mail: chatpon@yahoo.com

Mario T. Tabucanon
Industrial Engineering and Management Program, School of Advanced Technologies
Asian Institute of Technology, P.O. Box 4, Klong Luang, Pathumthani 12120, Thailand
Tel: +66-2-524-5002, E-mail: mtt@ait.ac.th, provost@ait.ac.th

Nguyen Van Hop
Industrial Engineering Program, Sirindhorn International Institute of Technology
Thammasat University, Pathumthani 12121, Thailand

Abstract. This paper presents new scheduling heuristics, namely Mean Progressive Weighted Tardiness Estimator (MPWT) Heuristic Method and modified priority rules with sequence-dependent setup times consideration. These are designed to solve job shop scheduling problems with new performance measures – progressive weighted tardiness penalties. More realistic constraints, which are inter-machine overlapping sequence-dependent setup times, are considered. In real production environments, inter-machine overlapping sequence-dependent setups are significant. Therefore, modified scheduling generation algorithms of active and nondelay schedules for job shop problems with inter-machine overlapping sequence-dependent setup times are proposed in this paper. In addition, new customer-based measures of performance, which are total earliness and progressive weighted tardiness, and total progressive weighted tardiness, are proposed. The objective of the first experiment is to compare the proposed priority rules with the consideration of sequence-dependent setup times and the standard priority rules without setup times consideration. The results indicate that the proposed priority rules with setup times consideration are superior to the standard priority rules without the consideration of setup times. From the second experiment and the third experiment to compare the proposed MPWT heuristic approach with the efficient priority rules with setup times consideration, the MPWT heuristic method is significantly superior to the Batched Apparent Tardiness Cost with Sequence-dependent Setups (BATCS) rule, and other priority rules based on total earliness and progressive weighted tardiness, and total earliness and tardiness.

Keywords: job shop scheduling problems, progressive weighted tardiness penalties, inter-machine overlapping setup times, sequence-dependent setup times

1. INTRODUCTION

The objective of this paper is to solve job shop scheduling problems with new constraints, which are inter-machine overlapping setup times. To reduce machine idle time, machine setups of successive operations of the same batch of parts are initiated simultaneously. For instance, let the batch size of a part be 100 and let the manufacturing processes of this part require two operations. The machine setup times of the second operation

† : Corresponding Author

can be initiated after only a fraction of the 100 parts has completed the first operation. These new realistic constraints in the job shop scheduling problems are called "inter-machine overlapping setup times". For solving the realistic job shop scheduling problems, the inter-machine overlapping setup times are sequence-dependent. Problems of sequence-dependent setup times often arise in production settings where the setup times are significant. For instance, at a production facility where paint is manufactured, the setup time incurred for cleaning machines depends on both the color being removed and the color for which it is being prepared. Similarly, in the plastic industry (Franca *et al*. 1996), items of different colors are typically assigned to different extrusion machines. When a color change is required, it takes a certain amount of time until the extruded plastic reaches the desired color. The setup times in this case depend on the sequence of jobs. Such problems are also common in the soft drink beverage industry where the manufacturing lines have to go through major setups when changing from filling glass bottles to soda cans. Similar examples can be found in the chemical and automotive parts manufacturing industries.

New measures of performance, which are total progressive weighted tardiness, and total earliness and progressssive weighted tardiness, are proposed in this paper. The new performance measures focus on customer satisfaction. The repetitive tardiness problems of the same customer lead to progressive weighted tardiness penalties. This issue will be addressed in section 4, and used in experiments in Section 7, Section 9 and Section 10. The objective of the first experiment in Section 7 is to determine the effect of sequence-dependent setup time consideration in the priority rules. The second experiment, to compare the Mean Progressive Weighted Tardiness Estimator (MPWT) heuristic method with BATCS, SMST and LWKRS, is conducted in Section 9. In Section 10, the third experiment is conducted to compare the MPWT heuristic method with other efficient heuristics based on the real scheduling data of a case study. The conclusions are drawn in Section 11.

2. LITERATURE REVIEW

Most of the research work performed on machine scheduling has not considered sequence-dependent setup times between jobs. In such cases, the setup times are assumed to be sequence independent and are considered to be part of the job direct processing times. Moreover, no research regarding job shop scheduling problems with new inter-machine overlapping setup time constraints, has been found in the literature. These production scheduling problems without considering inter-machine overlapping sequence-dependent setups could lead to unrealistic job shop scheduling problems.

Some research has been done that accounts for sequence dependency of job setup times along with due date considerations. Firstly, researchers paid attention to single-machine problems with sequence-dependent setups (see Picard and Queyranne 1978, Monma and Potts 1989, Uzsoy 1991, and Uzsoy 1992). The two problems of minimizing the maximum completion time for a sequence of jobs, and minimizing the maximum tardiness for jobs with a common due date, in the presence of sequence dependence, are equivalent to the famous travelling salesman problem (TSP). Monma and Potts (1989) examined various scheduling models that included batch setup times, where the setup times between jobs in the same batch were considered to be zero. Dynamic programming algorithms were used for the single machine problem, where the objective was to minimize the maximum completion time, maximum lateness, total weighted completion time and the number of tardy jobs. Picard and Queyranne (1978) modelled the problem of scheduling jobs with setup times on a single machine as a time-dependent travelling salesman problem. They used a branch and bound algorithm for this model. Uzsoy et al. (1991) discussed minimizing the maximum lateness in the presence of precedence constraints and sequence dependency of jobs. Each job was considered to have its own due date. A neighborhood search algorithm that obtained local optimal solutions was presented along with a branch and bound algorithm to obtain optimal solutions. In a later paper, Uzsoy et al. (1992) examined the problems of minimizing maximum lateness with dynamic arrivals and minimizing the number of tardy jobs in the presence of sequence-dependent setup times between jobs.

Sequence-dependent setup times have been addressed in parallel machine problems (see Dearing and Anderson 1984, Sumichrast and Baker 1987, Monma and Potts 1989, Franca et al. 1996, Kurz and Askin 2001). Dearing and Anderson (1984) studied the problem of scheduling jobs with sequence-dependent setup times on identical parallel machines, where the objective was to minimize the total setup cost. They developed an integer linear programming model and solved it as an LP with rounding procedures to attain integer solutions. Sumichrast and Baker (1987) improved the quality of these solutions by implementing a heuristic procedure that solved a series of 0-1 integer subproblems. The models formulated by Dearing and Anderson, and Sumichrast and Baker allowed jobs to be split among machines. Monma and Potts (1989) studied the parallel machine problem with preemption for sequence dependency of job setup times, where the objectives were minimizing maximum completion time, maximum lateness and number of tardy jobs, and showed that all these problems are NP-hard. Franca et al. (1996) considered the problem of scheduling jobs with sequence-dependent setup times on identical parallel machines where the objective was to minimize

the makespan. They obtained near optimal solutions for this problem using heuristics. The makespan minimization problem of parallel machines with sequence-dependent setup times and possibly non-zero ready times was solved by Kurz and Askin (2001) using an integer programming approach.

In job shop scheduling problems with sequence-dependent setup times, exact algorithms, considered with many constraints, consume too much computational time, so it is more appropriate to use heuristics. Apparent Tardiness Cost (ATC) is developed by Vepsalainen and Morton (1987). The slack of ATC is local resource constrainted slack which takes into account the waiting time on downstream machines, and the decay function for the weight/processing time ratio is exponential rather than linear. Several generalizations of the ATC rule have been developed to take sequence-dependent setup times into account (Pinedo 1995, Jeong and Kim 1998). The Apparent Tardiness Cost with Sequence-dependent Setups (ATCS) rule was proposed to minimize the sum of the weighted tardiness with consideration of sequence-dependent setup times. This implies that the priority of any job j depends on the job just completed on the machine just freed. It is obvious that the ATCS rule combines the WSPT and SLACK or MS rules in a single priority index. The rule calculates the index of job j succeeding job l at time t has competed its processing on the machine as

$$ATCS_{lj}(t) = \frac{w_j}{p_j}\exp\left(-\frac{\max\left(d_j - p_j - t, 0\right)}{k_1 p_{avg}}\right)\exp\left(-\frac{s_{lj}}{k_2 s_{avg}}\right) \quad (1)$$

where p_j is the processing time of job j, d_j is the due date of job j, w_j is the weight or tardiness penalty of job j, p_{avg} is the average of the processing times of jobs remaining to be scheduled, s_{lj} is the sequence-dependent setup time dependent on both the preceding job l and the succeeding job j, s_{avg} is the average of the setup times of jobs remaining to be scheduled, k_1 is the due date related scaling parameter, and k_2 is the setup time related scaling parameter. The apparent tardiness cost based rules have been presented in the papers of Jayamohan and Rajendran (2000), Thiagarajan and Rajendran (2003), and Balasubramanian, Monch, Fowler and Pfund (2004). Balasubramanian *et al*. (2004) recently presented the batched apparent tardiness cost (BATC). The jobs in each family are ordered in decreasing order of their ATC indices, and a batch of each customer is formed per family. In order to schedule one of these, the batched apparent tardiness cost (BATC) rule is used: at time t, for all the batches the following index is calculated

$$BATC_{xj} = \sum_{j \in B_{xj}} \frac{w_j}{p_j}\exp\left(-\frac{\max\left(d_j - p_j - t, 0\right)}{k\ p_{avg}}\right) \quad (2)$$

where $BATC_{xj}$ is the BATC index for batch x of family j. The batch with the highest BATC index is scheduled. In the paper of Balasubramanian *et al*. (2004), it was observed that the BATC rule provided a good solution in a relatively short amount of time even for the large-sized problems.

The BATC rule in the paper of Balasubramanian *et al*. (2004) has been generalized to be the batched apparent tardiness cost with sequence-dependent setups (BATCS) as follows.

$$BATCS_{(l)xj} = \sum_{j \in B_{xj}} \frac{w_j}{p_j}\exp\left(-\frac{\max\left(d_j - p_j - t, 0\right)}{k_1 p_{avg}}\right)\exp\left(-\frac{s_{lj}}{k_2 s_{avg}}\right) \quad (3)$$

The BATCS rule is used in the experiment to compare with the new MPWT heuristic method in Section 9 and Section 10.

3. PROBLEM STATEMENT

In this paper, job shop scheduling problems with total earliness and progressive weighted tardiness and inter-machine overlapping sequence-dependent setups are considered. The following questions arise:

- According to the more complex production environment, when the inter-machine overlapping sequence-dependent setups have to be considered in the job shop scheduling problems and the exact algorithm takes too much computational time to solve the problems, priority rules should be used. In the past, standard efficient priority rules – SPT (Shortest Processing Time), STPT (Shortest Total Processing Time), MWKR (Most Work Remaining) rules and others included the setup times in the direct processing times. When the inter-machine overlapping sequence-dependent setup times are significant and have to be taken into account, the modified standard priority rules are proposed in this paper. What are the experimental results when these proposed modified priority rules with setup times consideration are compared with the classical priority rules without consideration of setup times?
- How can we solve the new performance measures, which are progressive weighted tardiness penalties, effectively (with the lower values of the new measures of performance compared with other heuristics) and efficiently (with less computational time) in more complicated problems when inter-machine overlapping sequence-dependent setups are considered? In other words, a new scheduling heuristic method should be developed to determine a complete schedule with the lowest value of the new performance measure.

This paper proposes new performance measures as follows: (i) total earliness and progressive weighted tardiness and (ii) total progressive weighted tardiness. The new performance measures have emphasis on customer satisfaction.

4. TOTAL EARLINESS AND PROGRESSIVE WEIGHTED TARDINESS: A NEW MEASURE OF PERFORMANCE

Total earliness and progressive weighted tardiness is a new customer-based measure of performance. The repetitive penalties increase at a progressive rate, depending on how many times late delivery of finished goods to each customer occurs. In practice, when a manufacturer receives a customer order, there is a contract stipulating the desired types of products separated in batches, the due date of each batch, and the tardiness penalty. In a Just-In-Time production system, the customer has emphasis on tardiness problem. The reason is that it leads to a customer's material shortage problem, and has an effect on customer's tardiness problem. Therefore, the customer often specifies the progressive tardiness penalties. Although the progressive rates of progressive weighted tardiness penalties of the different customers are different; practically in most contracts with the same customer, constant progressive rates are stipulated in order to make the contracts not too complicated to understand and to reduce the complexity of the penalty computation. Therefore, the progressive rate of job for the same customer is assumed to be constant. In each job order of the same customer, the set of jobs is separated into several work pieces, each of which has its due date and associated penalty. The new performance measure based on the progressive weighted tardiness is stated as follows.

The penalty cost for the first time a job belonging to the same customer is tardy, is:

$$f_1(S) = \beta_{ij} \max\{0, c_{ij} - d_{ij}\} \quad \text{or} \quad f_1(S) = \beta_{ij} T_{ij} \tag{4}$$

where

c_{ij} = completion time of job j of customer i,
d_{ij} = due date of job j of customer i,
R_i = constant progressive tardiness rate of customer i (the progressive penalty rate is greater or equal to zero, and if the progressive penalty rate is equal to zero, the tardiness penalty of a job for the same customer is not increased; otherwise, it is increased by R_i *100% when jobs for the same customer are repeatedly late),
T_{ij} = tardiness of job j of customer i,
and β_{ij} = unit tardiness penalty for job j of customer i.

An amount of weighted tardiness penalty $f_2(S)$ is the second tardy job for the same customer. Hence, at a constant progressive tardiness penalty rate of R_i per time, the second late job for the same customer i, which causes the tardiness penalty increased by R_i *100%, will have the progressive weighted tardiness formulated as follows:

$$f_2(S) = f_1(S) + f_1(S) * R_i \tag{5}$$

$$f_2(S) = f_1(S)\left[1 + R_i\right] \tag{6}$$

At the third time of tardiness for the same customer, the amount of penalty accumulated, $f_3(S)$, will be equal to the amount accumulated after the second time of tardiness plus the additional penalty R_i*100% from the second time. Thus:

$$f_3(S) = f_2(S) + f_2(S) * R_i \tag{7}$$

$$= f_1(s)\left[1 + R\right] + f_1(s)\left[1 + R_i\right] * R_i \tag{8}$$

$$f_3(S) = f_1(s)\left[1 + R_i\right]^2 \tag{9}$$

$$f_3(S) = \beta_{ij} T_{ij}\left[1 + R_i\right]^2 \tag{10}$$

It is evident, by mathematical induction, that the formula can be generalized for n times of tardiness as:

$$f_n(S) = f_1(s)\left[1 + R\right]^{n-1} = \beta_{ij} T_{ij}\left[1 + R_i\right]^{n-1} \tag{11}$$

Hence, the total progressive weighted tardiness is:

$$F(S) = \sum_i \sum_j \beta_{ij} T_{ij}\left[1 + R_i\right]^{n-1} \tag{12}$$

Therefore, the objective function, which is total earliness and weighted progressive tardiness can be stated as:

Min

$$H(S) = \sum_i \sum_j \left(E_{ij} + \beta_{ij} T_{ij}\left[1 + R_i\right]^{n-1}\right) \tag{13}$$

To illustrate the calculation steps of the total earli-ess and progressive weighted tardiness, consider the ten-job, three-customer, five-machine problem described in Table 1 and Table 2. For example, the beginning time of the scheduling period is 1 November 2003. The Gantt chart, which is the graphic result of the complete schedule, is shown in Figure 1.

Table 1. Job details of the numerical example

Job Name	Due Time	Customer	Weight or Tardiness Penalty	Progressive Rate
JOB1	08:00	3	1	0.473
JOB2	09:00	1	5	0.7107
JOB3	09:00	1	10	0.7107
JOB4	08:30	2	6	0.9201
JOB5	08:30	2	4	0.9201
JOB6	08:00	3	9	0.473
JOB7	09:00	1	3	0.7107
JOB8	09:00	1	7	0.7107
JOB9	08:30	2	2	0.9201
JOB10	08:00	3	8	0.473

Table 2. Complete schedule – the output of the numerical example

Customer Name	Job Name	Machine ID	Due Time	Job Completion Time
1	JOB2	MCID3	09:00	1/11/2003 9:00
1	JOB8	MCID1	09:00	1/11/2003 9:35
1	JOB3	MCID5	09:00	1/11/2003 10:33
1	JOB7	MCID3	09:00	1/11/2003 10:59
2	JOB5	MCID5	08:30	1/11/2003 9:10
2	JOB4	MCID5	08:30	1/11/2003 10:06
2	JOB9	MCID5	08:30	1/11/2003 10:47
3	JOB10	MCID1	08:00	1/11/2003 10:12
3	JOB6	MCID4	08:00	1/11/2003 10:30
3	JOB1	MCID1	08:00	1/11/2003 11:07

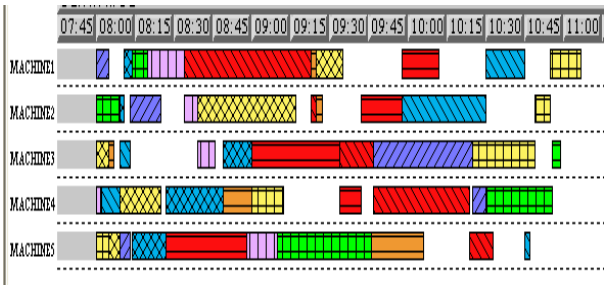


Figure 1. Gantt chart illustrating the complete schedule of the numerical example

The steps of the calculation of the total earliness and progressive weighted tardiness are presented as follows.

For CUSTOMER 1,
- Job 2 is not a tardy job.
  Due Date = 01-Nov-2003 09:00, Job Completion Time = 01 Nov-2003 09:00
- Job 8 is the first tardy job of CUSTOMER 1.
  Due Date =01-Nov-2003 09:00, Job Completion Time = 01-Nov-2003 09:35
n = 1, Progressive Rate = 0.7107
  Weight or Tardiness Penalty = 7, Tardiness = 35
  Progressive Weighted Tardiness =35*7 = 245
- Job 3 is the second tardy job of CUSTOMER 1.
  Due Date =01-Nov-2003 09:00, Job Completion Time = 01-Nov-2003 10:33
n = 2, Progressive Rate = 0.7107, Tardiness = 93 minutes
  Weight or Tardiness Penalty = 10
  Progressive Weighted Tardiness = $93*10*(1+0.7107)^{2-1} = 1590.951$
- Job 7 is the third tardy job of CUSTOMER 1.
  Due Date =01-Nov-2003 09:00, Job Completion Time = 01-Nov-2003 10:59
n = 3, Progressive Rate = 0.7107,
  Tardiness = 119 minutes
  Weight or Tardiness Penalty = 3
  Progressive Weighted Tardiness
  $= 119*3*(1+0.7107)^{3-1} = 1044.759$

For CUSTOMER 2,
- Job 5 is the first tardy job of CUSTOMER 2.
  Due Date = 01-Nov-2003 2001 08:30,
  Job Completion
  Time = 01-Nov-2003 09:10
n = 1, Progressive Rate = 0.9201, Tardiness = 40 minutes
  Weight or Tardiness Penalty = 4
  Progressive Weighted Tardiness = 160
- Job 4 is the second tardy job of CUSTOMER 2.
  Due Date =01-Nov-2003 08:30, Job Completion Time =01-Nov-2003 10:06
n = 2, Progressive Rate = 0.9201, Tardiness = 96 minutes
  Weight or Tardiness Penalty = 6
  Progressive Weighted Tardiness
  $= 96*6*(1+0.9201)^{2-1} = 1105.978$
- Job 9 is the third tardy job of CUSTOMER 2.
  Due Date =01-Nov-2003 08:30, Job Completion Time = 01-Nov-2003 10:47
n = 3, Progressive Rate = 0.9201,
  Weight or Tardiness Penalty = 2
  Progressive Weighted Tardiness = 1010.179

For CUSTOMER 3,
- Job 10 is the first tardy job of CUSTOMER 3.
  Due Date =01-Nov-2003 08:00, Job Completion Time = 01-Nov-2003 10:12
n = 1, Progressive Rate = 0.473
  Weight or Tardiness Penalty = 8
  Progressive Weighted Tardiness = 1056
- Job 6 is the second tardy job of CUSTOMER 3.
  Due Date =01-Nov-2003 08:00, Job Completion Time = 01-Nov-2003 10:30
n = 2, Progressive Rate = 0.473
  Tardiness Penalty Weight = 9
  Progressive Weighted Tardiness = 1988.55
- Job 1 is the third tardy job of CUSTOMER 3.
  Due Date =01-Nov-2003 08:00, Job Completion Time = 01-Nov-2003 11:07
n = 3, Progressive Rate = 0.473, Tardiness = 187 minutes
  Weight or Tardiness Penalty = 1
  Progressive Weighted Tardiness
  $= 187*1*(1+0.473)^{3-1} = 405.739323$

The Total Progressive Weighted Tardiness = 245 + 1590.951 + 1044.759 + 160 + 1105.98 + 1010.18 + 1056 + 1988.55 + 405.74 = 8607.16. Since there is no early job, total earliness = 0, the total earliness and progressive weighted tardiness is equal to 8,607.16.

5. INTER-MACHINE OVERLAPPING SEQUENCE-DEPENDENT SETUP TIMES

The new characteristic of job shop scheduling proposed in this paper is inter-machine overlapping sequen-

ce-dependent setup time of a job. In this case, the job is considered as a batch of n parts. To reduce machine idle time and job flow time, machine setups of successive operations of the same batch of parts are initiated after few parts of the same batch in the preceding operation have been finished. Inter-machine overlapping sequence-dependent setup times are defined as the amount of time between the earliest time required to start inter-machine overlapping setup, which follows the precedence constraints of job direct processing times and the capacity constraints, and the end of the inter-machine overlapping setup. Let job (i, j, k) be the minimum transport batch of operation j of product type i on machine k, and it cannot be separated into smaller batches, so the precedence constraints are adopted. Therefore, the job direct processing times of the minimum transport batch cannot be overlapped, and only setup times can be overlapped. Normally, the "batch" and "job" that are mentioned in this paper mean the minimum transport batch. There are two cases, as shown below:

Case A. When the batch direct processing time of the preceding operation is shorter than the sequence-dependent setup time of the succeeding operation. The machine setup of the succeeding operation can start as soon as one part of the same batch in the preceding operation is finished (direct processing time of one part is assumed to be much smaller than direct processing time of the minimum transport batch). Therefore, the earliest be-ginning time of machine setup is given by:

Inter-machine Overlapping Start Setup Time for the succeeding operation = Finish Setup Time for the preceding operation.

Case B. When the batch direct processing time of the preceding operation is longer than the sequence-dependent setup time of the succeeding operation; due to the precedence constraints of the minimum transport batch, which is called "job", direct processing times cannot be overlapped. Thus, the earliest beginning time of machine setup is given by:

Inter-machine overlapping Start Setup Time for the succeeding operation = *Non-intermachine overlapping Earliest Start Setup Time – Setup Time*.

These two cases are described in Figure 2.

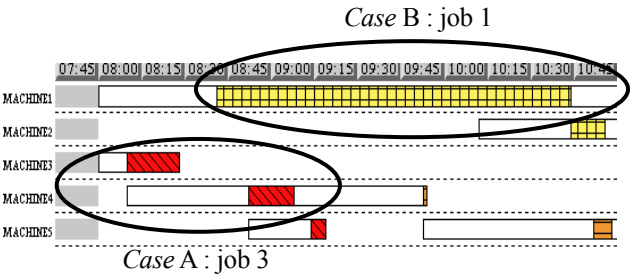


Figure 2. Gantt chart of the inter-machine overlapping sequence-dependent setup time in *Case* A and *Case* B

The calculation steps are as follows.

Step1 Find the *Start Setup* Time according to precedence constraints. Find the *Start Setup* Time, which is the beginning of non-intermachine overlapping setup time, by comparing the finish time of job i and the finish time of the machine (the time machine status is changed from busy to idle), which job i works on. If the finish time of job i is greater than the finish time of the machine that job i works on, then set *Start Setup Time* = Finish time of job i, else set *Start Setup Time* = Finish time of the machine that job i works on.

Step 2 Determine the inter-machine overlapping *Start Setup Time*. Determine *Setup Time* for the considered schedulable operation of job i from the sequence-dependent setup time database. Calculate inter-machine overlapping *Start Setup Time* = non-inter-machine overlapping *Start Setup Time* – Setup Time. Find the *Finish Setup Time* of the preceding operation of job i.

Step 3 Compare *Start Setup Time* with *Finish Setup Time* of the preceding operation of job i. Compare inter-machine overlapping *Start Setup Time* and *Fini-sh Setup Time* of the preceding operation of job i. If *Start Setup Time* is less than *Finish Setup Time* of the preceding operation of job i, then set *Start Setup Time* = *Finish Setup Time* of the preceding operation of job i.

Step 4 Determine the finish time of inter-machine overlapping setups. Determine the finish time of inter-machine overlapping setups by comparing between *Start Setup Time* and Finish time of the preceding operation of job i subtracted by Setup time. If *Start Setup Time* is less than Finish time of the preceding operation subtracted by Setup time, then set *Start Setup Time* = Finish time of the preceding opera-tion – Setup time.

Step 5 Check machine capacity constraints. Due to machine capacity constraints, if there are some jobs still on the machine, the batch inter-machine overlapping setups cannot begin. Therefore, the machine capacity constraints are checked by comparing *Start Setup Time* with Finish time of the machine that job i works on. If inter-machine overlapping Start Setup Time is less than Finish time of machine that job i works on, then set *Start Setup Time* = Finish time of the machine that job i works on.

The numerical examples for *Case* A and *Case* B, illustrated in Figure 2, are as follows.

Case A. There are three operations of the red job (job 3). Consider the second operation of the red job.

Let *Sequence-dependent Setup Time* of the second operation of the red job = 42 minutes and *Non-inter-*

machine overlapping Start Setup Time = 1 February 2004 8:28 a.m.

Start Setup Time = Non-intermachine overlapping Start Setup Time – Setup Time.

Start Setup Time = 1 February 2004 8:28 a.m. – 42 minutes = 1 February 2004 7:46 a.m.

Finish Setup Time for the preceding operation (the first operation of the red job) = 1 February 2004 8:10 a.m.

Start Setup Time for the second operation of the red job is earlier than *Finish Setup Time* for the preceding operation (the first operation of the red job), then set *Start Setup Time = Finish Setup Time* for the preceding operation = 1 February 2004 8:10 a.m.

Therefore, the end of setup time for the second operation is 1 February 2004 8:52 a.m. After checking the capacity constraints, the inter-machine overlapping sequence-dependent setup starts at 8:10 a.m. and finishes at 8:52 a.m.

Case B. There are two operations of the yellow job (job 1). Consider the second operation of the yellow job.

Let *Sequence-dependent Setup time* of the second operation of the yellow job = 32 minutes and *Non-intermachine overlapping Start Setup Time* = 1 February 2004 10:44 a.m.

Start Setup Time = Non-intermachine overlapping Start Setup Time – Setup Time.

Start Setup Time = 1 February 2004 10:44 a.m. – 32 minutes = 1 February 2004 10:12 a.m.

Finish Setup Time for the preceding operation (the first operation of the yellow job) = 1 February 2004 8:41 a.m.

Start Setup Time for the second operation of the yellow job is not earlier than Finish Setup Time for the preceding operation (the first operation of the yellow job), then *Start Setup Time* for the second operation is 1 February 2004 10:12 a.m.

Therefore, the end of setup time for the second operation is 1 February 2004 10:44 a.m. After checking the capacity constraints, the inter-machine overlapping sequence-dependent setup starts at 10:12 a.m. and finishes at 10:44 a.m.

6. MODIFIED PRIORITY RULES WITH SEQUENCE-DEPENDENT SETUP TIMES

When the inter-machine overlapping sequence-dependent setup times are significant, the standard priority rules should be modified. In this paper, we propose five modified priority rules. The details and numerical examples of the five modified priority rules generated by non-delay scheduling algorithm are as follows.

Consider the three-job, three operation, three-machine problem shown in Tables 3, 4, 5, 6, and 7. Let the beginning time of job shop scheduling be 01 February 2004 8:00 a.m.

Table 3. Routing table of the numerical example

Job	Operation 1	Operation 2	Operation 3
1	Machine 3	Machine 2	Machine 1
2	Machine 2	Machine 3	Machine 1
3	Machine 1	Machine 2	Machine 3

Table 4. Processing times table of the numerical example

Job	Processing T1	Processing T2	Processing T3
1	3.9	7.81	21.06
2	3.38	29.87	1.99
3	2.94	12.29	3.19

Table 5. Sequence-dependent setup times table for Machine 1

From \ To	Job 1	Job 2	Job 3
Job 1	-	56	19
Job 2	10	-	33
Job 3	58	13	-

Table 6. Sequence-dependent setup times table for Machine 2

From \ To	Job 1	Job 2	Job 3
Job 1	-	15	21
Job 2	26	-	57
Job 3	54	57	-

Table 7. Sequence-dependent setup times table for Machine 3

From \ To	Job 1	Job 2	Job 3
Job 1	-	36	18
Job 2	43	-	45
Job 3	42	29	-

6.1 Least Work Remaining with sequence-dependent setup times (LWKRS) Rule

The work remaining values of the considered operations in the set of active and nondelay schedules are used in the standard LWKR rule to solve the conflicting operations. However, for the job shop scheduling problems with inter-machine overlapping sequence-dependent setup times, the sequence-dependent setups should be considered. Therefore, the priority index of the new modified rule is the summation of sequence-dependent setup time and the work remaining.

The calculation steps of LWKRS are as follows.

Step 1: Set the initial value of *RetValue*, which is the initial value of work remaining, to be the possible

maximum value.
Step 2: Do the loop until all of the conflicting operations in the set of active or nondelay schedules have been considered.
Step 3: Calculate the work remaining of the considered conflicting operation.
Step 4: Determine the sequence-dependent setup time of the considered conflicting operation by looking for the setup time in the scheduling database.
Step 5: Set the *Work remaining* = *Work remaining time* + *Sequence-dependent setup time*.
Step 6: Compare the *RetValue* and the *Work remaining* calculated in Step 5. If the *Work remaining* is lower than *RetValue*, then go to Step 7, otherwise go to Step 8.
Step 7: Set the value of *RetValue* = *Work remaining*. Then go to Step 8.
Step 8: Shift to the job set of the next conflicting operations.

The numerical example of scheduling data from Table 3 to Table 7 using the nondelay scheduling generation algorithm with the LWKRS rule is shown below.

Loop : 1
PS : {Empty Set}
Sigma (the earliest start time of each operation) :
(1,1,3 := 01-Feb-04 08:00:00),
(2,1,2 := 01-Feb-04 08:00:00),
(3,1,1 := 01-Feb-04 08:00:00),
Sigma* (the smallest value of Sigma) :
01-Feb-04 08:00:00
M* : 3,2,1
Nondelay schedulable operations :
{(1,1,3),(2,1,2),(3,1,1)}
LWKRS :
LWKRS(1,1,3) = 0+33=33; LWKRS(2,1,2) =
0+35=35; LWKRS(3,1,1) = 0+18=18
Selected Job : (3,1,1)
Loop : 2
PS : {(3,1,1)}
Sigma : (1,1,3 := 01-Feb-04 08:00:00),
(2,1,2 := 01-Feb-04 08:00:00),
(3,2,2 := 01-Feb-04 08:03:00),
Sigma* : 01-Feb-04 08:00:00
M* : 3,2
Nondelay schedulable operations : {(1,1,3),(2,1,2)}
LWKRS : LWKRS(1,1,3) = 0+33=33;
LWKRS(2,1,2) = 0+35=35
Selected Job : (1,1,3)
Loop : 3
PS : {(3,1,1),(1,1,3)}
Sigma : (1,2,2 := 01-Feb-04 08:04:00),
(2,1,2 := 01-Feb-04 08:00:00),
(3,2,2 := 01-Feb-04 08:03:00),
Sigma* : 01-Feb-04 08:00:00
M* : 2
Nondelay schedulable operations : {(2,1,2)}
LWKRS : LWKRS(2,1,2) = 0+35=35
Selected Job : (2,1,2)
Loop : 4
PS : {(3,1,1),(1,1,3),(2,1,2)}
Sigma : (1,2,2 := 01-Feb-04 08:03:00),
(2,2,3 := 01-Feb-04 08:04:00),
(3,2,2 := 01-Feb-04 08:03:00),
Sigma* : 01-Feb-04 08:03:00
M* : 2
Nondelay schedulable operations : {(1,2,2),(3,2,2)}
LWKRS : LWKRS(1,2,2) = 26+29=55;
LWKRS(3,2,2) = 57+15=72
Selected Job : (1,2,2)
Loop : 5
PS : {(3,1,1),(1,1,3),(2,1,2),(1,2,2)}
Sigma : (1,3,1 := 01-Feb-04 08:29:00),
(2,2,3 := 01-Feb-04 08:04:00),
(3,2,2 := 01-Feb-04 08:37:00),
Sigma* : 01-Feb-04 08:04:00
M* : 3
Nondelay schedulable operations : {(2,2,3)}
LWKRS : LWKRS(2,2,3) = 36+32=68
Selected Job : (2,2,3)
Loop : 6
PS : {(3,1,1),(1,1,3),(2,1,2),(1,2,2),(2,2,3)}
Sigma : (1,3,1 := 01-Feb-04 08:29:00),
(2,3,1 := 01-Feb-04 08:57:00),
(3,2,2 := 01-Feb-04 08:37:00),
Sigma* : 01-Feb-04 08:29:00
M* : 1
Nondelay schedulable operations : {(1,3,1)}
LWKRS : LWKRS(1,3,1) = 58+21=79
Selected Job : (1,3,1)
Loop : 7
PS : {(3,1,1),(1,1,3),(2,1,2),(1,2,2),(2,2,3),(1,3,1)}
Sigma : (2,3,1 := 01-Feb-04 09:48:00),
(3,2,2 := 01-Feb-04 08:37:00),
Sigma* : 01-Feb-04 08:37:00
M* : 2
Nondelay schedulable operations : {(3,2,2)}
LWKRS : LWKRS(3,2,2) = 21+15=36
Selected Job : (3,2,2)
Loop : 8
PS : {(3,1,1), (1,1,3), (2,1,2), (1,2,2), (2,2,3), (1,3,1),
(3,2,2)}
Sigma : (2,3,1 := 01-Feb-04 09:48:00),
(3,3,3 := 01-Feb-04 09:10:00),
Sigma* : 01-Feb-04 09:10:00
M* : 3
Nondelay schedulable operations : {(3,3,3)}
LWKRS : LWKRS(3,3,3) = 45+3=48
Selected Job : (3,3,3)

Loop : 9
PS : {(3,1,1), (1,1,3), (2,1,2), (1,2,2),(2,2,3),(1,3,1), (3,2,2), (3,3,3)}
Sigma : (2,3,1 := 01-Feb-04 09:48:00),
Sigma* : 01-Feb-04 09:48:00
M* : 1
Nondelay schedulable operations : {(2,3,1)
LWKRS : LWKRS(2,3,1) = 56+2=58
Selected Job : (2,3,1)
Finish :
Complete Schedule : {(3,1,1), (1,1,3), (2,1,2), (1,2,2), (2,2,3), (1,3,1), (3,2,2), (3,3,3), (2,3,1)}

6.2 Most Work Remaining with sequence-dependent setup times (MWKRS) Rule

As in LWKR, the values of the work remaining of conflicting operations in the set of active and nondelay schedules are used as the priority index in the standard MWKR rule. In contrast, the conflicting operation with the greatest value of the work remaining is selected. However, for the job shop scheduling problems with inter-machine overlapping sequence-dependent setup times, the sequence-dependent setups should be considered. Therefore, the priority index of the new modified rule is the summation of the sequence-dependent setup time and the work remaining.

The calculation steps of MWKRS are as follows.

Step 1 Set the initial value of RetValue, which is the initial value of work remaining, to be the possible minimum value.
Step 2 Do the loop until all of the conflicting operations in the set of active or nondelay schedules have been considered.
Step 3 Calculate the work remaining of the considered conflicting operation.
Step 4 Determine the sequence-dependent setup time of the considered conflicting operation by looking for the setup time in the scheduling database.
Step 5 Set the Work remaining = Work remaining time + Sequence-dependent setup time.
Step 6 Compare the RetValue and the Work remaining calculated in Step 5. If the Work remaining is greater than RetValue, then go to Step 7, otherwise go to Step 8.
Step 7 Set the value of RetValue = Work remaining. Then go to Step 8.
Step 8 Shift to the job set of the next conflicting operations.

The numerical example of scheduling data from Table 3 to Table 7 using the nondelay schedule generation algorithm with the MWKRS rule is presented as follows.

Loop : 1
PS : {Empty Set}
Sigma : (1,1,3 := 01-Feb-04 08:00:00),
(2,1,2 := 01-Feb-04 08:00:00),
(3,1,1 := 01-Feb-04 08:00:00),
Sigma* : 01-Feb-04 08:00:00
M* : 3,2,1
Nondelay schedulable operations: {(1,1,3), (2,1,2), (3,1,1)}
MWKRS : MWKRS(1,1,3) = 0+33=33;
MWKRS(2,1,2) = 0+35=35;
MWKRS(3,1,1) = 0+18=18
Selected Job : (2,1,2)
Loop : 2
PS : {(2,1,2)}
Sigma : (1,1,3 := 01-Feb-04 08:00:00),
(2,2,3 := 01-Feb-04 08:03:00),
(3,1,1 := 01-Feb-04 08:00:00),
Sigma* : 01-Feb-04 08:00:00
M* : 3,1
Nondelay schedulable operations: {(1,1,3), (3,1,1)}
MWKRS : MWKRS(1,1,3) = 0+33=33;
MWKRS(3,1,1) = 0+18=18
Selected Job : (1,1,3)
Loop : 3
PS : {(2,1,2), (1,1,3)}
Sigma : (1,2,2 := 01-Feb-04 08:03:00),
(2,2,3 := 01-Feb-04 08:04:00),
(3,1,1 := 01-Feb-04 08:00:00),
Sigma* : 01-Feb-04 08:00:00
M* : 1
Nondelay schedulable operations: {(3,1,1)}
MWKRS : MWKRS(3,1,1) = 0+18=18
Selected Job : (3,1,1)
Loop : 4
PS : {(2,1,2), (1,1,3), (3,1,1)}
Sigma : (1,2,2 := 01-Feb-04 08:03:00),
(2,2,3 := 01-Feb-04 08:04:00),
(3,2,2 := 01-Feb-04 08:03:00),
Sigma* : 01-Feb-04 08:03:00
M* : 2
Nondelay schedulable operations: {(1,2,2), (3,2,2)}
MWKRS : MWKRS(1,2,2) = 26+29=55;
MWKRS(3,2,2) = 57+15=72
Selected Job : (3,2,2)
Loop : 5
PS : {(2,1,2), (1,1,3), (3,1,1), (3,2,2)}
Sigma : (1,2,2 := 01-Feb-04 09:12:00),
(2,2,3 := 01-Feb- 01 08:04:00),
(3,3,3 := 01-Feb-04 09:00:00),
Sigma* : 01-Feb-04 08:04:00
M* : 3
Nondelay schedulable operations: {(2,2,3)}
MWKRS : MWKRS(2,2,3) = 36+32=68
Selected Job : (2,2,3)

Loop : 6
PS : {(2,1,2), (1,1,3), (3,1,1), (3,2,2), (2,2,3)}
Sigma : (1,2,2 := 01-Feb-04 09:12:00),
(2,3,1 := 01-Feb-04 08:57:00),
(3,3,3 := 01-Feb-04 09:10:00),
Sigma* : 01-Feb-04 08:57:00
M* : 1
Nondelay schedulable operations: {(2,3,1)}
MWKRS : MWKRS(2,3,1) = 13+2=15
Selected Job : (2,3,1)
Loop : 7
PS : {(2,1,2), (1,1,3), (3,1,1), (3,2,2), (2,2,3), (2,3,1)}
Sigma : (1,2,2 := 01-Feb-04 09:12:00),
(3,3,3 := 01-Feb-04 09:10:00),
Sigma* : 01-Feb-04 09:10:00
M* : 3
Nondelay schedulable operations: {(3,3,3)}
MWKRS : MWKRS(3,3,3) = 45+3=48
Selected Job : (3,3,3)
Loop : 8
PS : {(2,1,2), (1,1,3), (3,1,1), (3,2,2), (2,2,3), (2,3,1), (3,3,3)}
Sigma : (1,2,2 := 01-Feb-04 09:12:00),
Sigma* : 01-Feb-04 09:12:00
M* : 2
Nondelay schedulable operations: {(1,2,2)}
MWKRS : MWKRS(1,2,2) = 54+29=83
Selected Job : (1,2,2)
Loop : 9
PS : {(2,1,2), (1,1,3), (3,1,1), (3,2,2), (2,2,3), (2,3,1), (3,3,3), (1,2,2)}
Sigma : (1,3,1 := 01-Feb-04 10:06:00),
Sigma* : 01-Feb-04 10:06:00
M* : 1
Nondelay schedulable operations: {(1,3,1)}
MWKRS : MWKRS(1,3,1) = 10+21=31
Selected Job : (1,3,1)
Finish :
Complete Schedule : {(2,1,2), (1,1,3), (3,1,1), (3,2,2), (2,2,3), (2,3,1), (3,3,3), (1,2,2), (1,3,1)}

6.3 Shortest Total Sequence-dependent Setup and Processing Times (SSPT) Rule

According to the standard SPT (Shortest Processing Time), to solve the conflicting operations in the set of active and nondelay schedules, the operation with the shortest processing time is selected for processing next. Sequence-dependent setup times should be taken into account in the popular SPT rule, namely SSPT rule.

The calculation steps of SSPT are as follows.

Step 1 Set the initial value of RetValue, which is the initial value of processing time of the conflicting operation, to be the possible maximum value.
Step 2 Do the loop until all of the conflicting operations in the set of active or nondelay schedules have been considered.
Step 3 Determine the sequence-dependent setup time of the considered conflicting operation by looking for the setup time in the scheduling database.
Step 4 Set the *Processing time = Sequence-dependent setup time + Processing time*.
Step 5 Compare the *RetValue* and the *Processing time* calculated in Step 4. If the *Processing time* is lower than *RetValue*, then go to Step 6, otherwise go to Step 7.
Step 6 Set the value of *RetValue = Processing time*. Then go to Step 7.
Step 7 Shift to the job set of the next conflicting operations.

The numerical example of scheduling data in from Table 3 to Table 7 using SSPT rule to solve the conflicting operation in the set of nondelay schedules is shown below.

Loop : 1
PS : {Empty Set}
Sigma : (1,1,3 := 01-Feb-04 08:00:00),
(2,1,2 := 01-Feb-04 08:00:00),
(3,1,1 := 01-Feb-04 08:00:00),
Sigma* : 01-Feb-04 08:00:00
M* : 3,2,1
Nondelay schedulable operations :
{(1,1,3), (2,1,2), (3,1,1)}
SSPT : SSPT(1,1,3) = 0+4=4;
SSPT(2,1,2) = 0+3=3;
SSPT (3,1,1) = 0+3=3
Selected Job : (2,1,2)
Loop : 2
PS : {(2,1,2)}
Sigma : (1,1,3 := 01-Feb-04 08:00:00),
(2,2,3 := 01-Feb-04 08:03:00),
(3,1,1 := 01-Feb-04 08:00:00),
Sigma* : 01-Feb-04 08:00:00
M* : 3,1
Nondelay schedulable operations : {(1,1,3), (3,1,1)}
SSPT : SSPT(1,1,3) = 0+4=4;
SSPT(3,1,1) = 0+3=3
Selected Job : (3,1,1)
Loop : 3
PS : {(2,1,2), (3,1,1)}
Sigma : (1,1,3 := 01-Feb-04 08:00:00),
(2,2,3 := 01-Feb-04 08:03:00),
(3,2,2 := 01-Feb-04 08:03:00),
Sigma* : 01-Feb-04 08:00:00
M* : 3
Nondelay schedulable operations : {(1,1,3)}
SSPT : SSPT(1,1,3) = 0+4=4

Selected Job : (1,1,3)
Loop : 4
PS : {(2,1,2), (3,1,1), (1,1,3)}
Sigma : (1,2,2 := 01-Feb-04 08:03:00),
(2,2,3 := 01-Feb-04 08:04:00),
(3,2,2 := 01-Feb-04 08:03:00),
Sigma* : 01-Feb-04 08:03:00
M* : 2
Nondelay schedulable operations : {(1,2,2),(3,2,2)}
SSPT : SSPT(1,2,2) = 26+8=34;
SSPT(3,2,2) = 57+12=69
Selected Job : (1,2,2)
Loop : 5
PS : {(2,1,2), (3,1,1), (1,1,3), (1,2,2)}
Sigma : (1,3,1 := 01-Feb-04 08:29:00),
(2,2,3 := 01-Feb-04 08:04:00),
(3,2,2 := 01-Feb-04 08:37:00),
Sigma* : 01-Feb-04 08:04:00
M* : 3
Nondelay schedulable operations : {(2,2,3)}
SSPT : SSPT(2,2,3) = 36+30=66
Selected Job : (2,2,3)
Loop : 6
PS : {(2,1,2), (3,1,1), (1,1,3), (1,2,2), (2,2,3)}
Sigma : (1,3,1 := 01-Feb-04 08:29:00),
(2,3,1 := 01-Feb-04 08:57:00),
(3,2,2 := 01-Feb-04 08:37:00),
Sigma* : 01-Feb-04 08:29:00
M* : 1
Nondelay schedulable operations : {(1,3,1)}
SSPT : SSPT(1,3,1) = 58+21=79
Selected Job : (1,3,1)
Loop : 7
PS : {(2,1,2), (3,1,1), (1,1,3), (1,2,2), (2,2,3), (1,3,1)}
Sigma : (2,3,1 := 01-Feb-04 09:48:00),
(3,2,2 := 01-Feb-04 08:37:00),
Sigma* : 01-Feb-04 08:37:00
M* : 2
Nondelay schedulable operations : {(3,2,2)}
SSPT : SSPT(3,2,2) = 21+12=33
Selected Job : (3,2,2)
Loop : 8
PS : {(2,1,2), (3,1,1), (1,1,3), (1,2,2), (2,2,3), (1,3,1), (3,2,2)}
Sigma : (2,3,1 := 01-Feb-04 09:48:00),
(3,3,3 := 01-Feb-04 09:10:00),
Sigma* : 01-Feb-04 09:10:00
M* : 3
Nondelay schedulable operations : {(3,3,3)}
SSPT : SSPT(3,3,3) = 45+3=48
Selected Job : (3,3,3)
Loop : 9
PS : {(2,1,2), (3,1,1), (1,1,3), (1,2,2), (2,2,3), (1,3,1), (3,2,2), (3,3,3)}
Sigma : (2,3,1 := 01-Feb-04 09:48:00),
Sigma* : 01-Feb-04 09:48:00
M* : 1
Nondelay schedulable operations : {(2,3,1)}
SSPT : SSPT(2,3,1) = 56+2=58
Selected Job : (2,3,1)
Finish :
Complete Schedule : {(2,1,2), (3,1,1), (1,1,3), (1,2,2), (2,2,3), (1,3,1), (3,2,2), (3,3,3), (2,3,1)}

6.4 Smallest Value Obtained by Multiplying Total Setup and Processing Times with Total Processing Time (SMST) Rule

The calculation steps of SMST are as follows.

Step 1 Set the initial value of RetValue to be the possible maximum value.
Step 2 Do the loop until all of the conflicting operations in the set of active or nondelay schedules have been considered.
Step 3 Determine the sequence-dependent setup time of the considered conflicting operation by looking for the setup time in the scheduling database.
Step 4 Set the *Processing time = Sequence-dependent setup time + Processing time*.
Step 5 Compare the *RetValue* and the *Total processing time * Processing time* calculated in Step 4. If the *Total processing time * Processing time* is lower than RetValue, then go to Step 6, otherwise go to Step 7.
Step 6 Set the value of *RetValue = Total processing time * Processing time*. Then go to Step 7.
Step 7 Shift to the job set of the next conflicting operations.

As in SSPT, the nondelay scheduling algorithm is the same; however, the conflicting operations in the set of nondelay schedules are solved as follows.

Loop : 1
PS : {Empty Set}
Sigma : (1,1,3 := 01-Feb-04 08:00:00),
(2,1,2 := 01-Feb-04 08:00:00),
(3,1,1 := 01-Feb-04 08:00:00),
Sigma* : 01-Feb-04 08:00:00
M* : 3,2,1
Nondelay schedulable operations :
{(1,1,3), (2,1,2), (3,1,1)}
SMST : SMST = (Sequence-dependent Setup time + Processing time)* Total processing time
SMST(1,1,3) = (0+4)*33=132;
SMST (2,1,2) = (0+3)*35=105;
SMST(3,1,1) = (0+3)*18=54
Selected Job : (3,1,1)
Loop : 2
PS : {(3,1,1)}

Sigma : (1,1,3 := 01-Feb-04 08:00:00),
(2,1,2 := 01-Feb-04 08:00:00),
(3,2,2 := 01-Feb-04 08:03:00),
Sigma* : 01-Feb-04 08:00:00
M* : 3,2
Nondelay schedulable operations : {(1,1,3),(2,1,2)}
SMST : SMST(1,1,3) = (0+4)*33=132;
SMST(2,1,2) = (0+3)*35=105
Selected Job : (2,1,2)
Do loop until a complete schedule is obtained.

6.5 Shortest Total Sequence-dependent Setup and Total Processing Times (SSTPT) Rule

From the standard Shortest Total Processing Time (STPT), the total processing time is used to solve the conflicting operations. The modified STPT, namely SSTPT, is proposed.

The calculation steps of SSTPT are as follows.

Step 1 Set the initial value of RetValue, which is the initial value of the job total processing time of the conflicting operation, to be the possible maximum value.

Step 2 Do the loop until all of the conflicting operations in the set of active or nondelay schedules have been considered.

Step 3 Determine the sequence-dependent setup time of the considered conflicting operation by looking for the setup time in the scheduling database.

Step 4 Set the *Total processing time* = *Sequence-dependent setup time* + *Total processing time*.

Step 5 Compare the *RetValue* and the *Total processing time* calculated in Step 4. If the *Total processing time* is lower than *RetValue*, then go to Step 6, otherwise go to Step 7.

Step 6 Set the value of RetValue = Total processing time. Then go to Step 7.

Step 7 Shift to the job set of the next conflicting operations.

As in SSPT and SMST, the nondelay scheduling algorithm is the same; however, the conflicting operations in the set of nondelay schedules are solved as follows.

Loop : 1
PS : {Empty Set}
Sigma : (1,1,3 := 01-Feb-04 08:00:00),
(2,1,2 := 01-Feb-04 08:00:00),
(3,1,1 := 01-Feb-04 08:00:00),
Sigma* : 01-Feb-04 08:00:00
M* : 3,2,1
Nondelay schedulable operations :
{(1,1,3), (2,1,2), (3,1,1)}
SSTPT : SSTPT(1,1,3) = 0+33=33;
SSTPT(2,1,2) = 0+35=35;
SSTPT(3,1,1) = 0+18=18
Selected Job : (3,1,1)
Loop : 2
PS : {(3,1,1)}
Sigma : (1,1,3 := 01-Feb-04 08:00:00),
(2,1,2 := 01-Feb-04 08:00:00),
(3,2,2 := 01-Feb-04 08:03:00),
Sigma* : 01-Feb-04 08:00:00
M* : 3,2
Nondelay schedulable operations :
{(1,1,3), (2,1,2)}
SSTPT : SSTPT(1,1,3) = 0+33=33;
SSTPT(2,1,2) = 0+35=35
Selected Job : (1,1,3)
Do loop until a complete schedule is obtained.

7. COMPARISON BETWEEN THE MODIFIED PRIORITY RULES WITH SETUP TIMES CONSIDERATION AND THE CLASSICAL PRIORITY RULES

The nondelay scheduling generation algorithms are used in this experiment. There are two important factors considered in this experiment as follows: (i) priority rules for solving conflicting operations in the set of nondelay schedules, and (ii) sequence-dependent setup time consideration (with or without setup times consideration). For the priority rules, there are five modified priority rules as mentioned in Section 6. Mean values of ten replications are determined to use in the comparison. The experimental results are shown in Figures 3, 4 and 5.

The meanings of Priority rule 1 to Priority rule 10 are

Rule 1 is LWKR	Rule 6 is LWKRS
Rule 2 is MWKR	Rule 7 is MWKRS
Rule 3 is SMT	Rule 8 is SMST
Rule 4 is SPT	Rule 9 is SSPT
Rule 5 is STPT	Rule 10 is SSTPT.

The three measures of performance are considered in this experiment as follows:

- Total earliness and progressive weighted tardiness (Figure 3)
- Total progressive weighted tardiness (Figure 4)
- Total earliness and tardiness (Figure 5)

From Figure 3 to Figure 5, the proposed priority rules with setup times consideration, which are LWKRS, MWKRS, SMST, SSPT, and SSTPT rules are superior to the classical priority rules, which are LWKR, MWKR, SMT, SPT, and STPT. The sequence-dependent setup time consideration has a significant effect on the experimented performance measures.

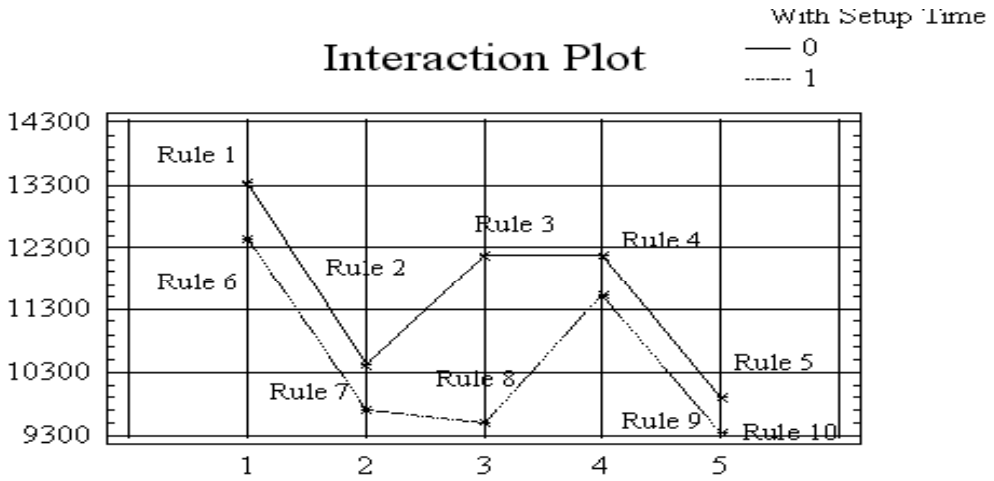


Figure 3. Priority rules with setup times comparison results based on total earliness and progressive weighted tardiness

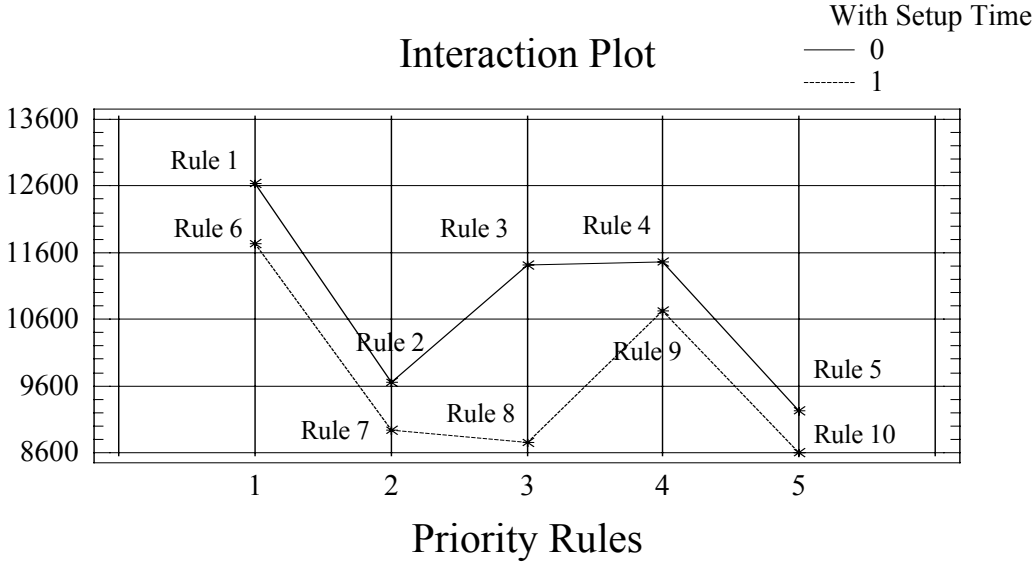


Figure 4. Priority rules with setup times comparison results based on total progressive weighted tardiness

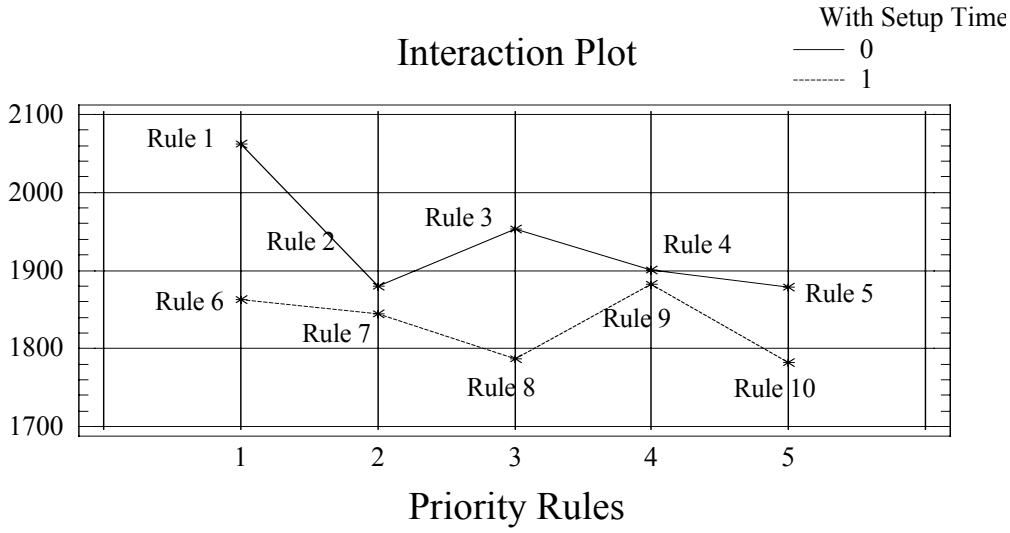


Figure 5. Priority rules with setup times comparison results based on total earliness and tardiness

Table 8. The comparison data of the priority rules with setup times consideration

Priority Rules	Total Progressive Weighted Tardiness	Total Earliness and Progressive Weighted Tardiness	Total Earliness and Tardiness
LWKR (Rule 1)	12628.5	13323.7	2062.3
LWKRS (Rule 6)	11735.3	12428	1862.2
Difference	893.2	895.7	200.1
% Difference	7.07%	6.72%	9.70%
MWKR (Rule 2)	9655.11	10412.4	1879.8
MWKRS (Rule 7)	8938.37	9693.72	1844.5
Difference	716.74	718.68	35.3
% Difference	7.42%	6.90%	1.88%
SMT (Rule 3)	11410.2	12166.4	1953.65
SMST (Rule 8)	8756.35	9509.45	1786.35
Difference	2653.85	2656.95	167.3
% Difference	23.26%	21.84%	8.56%
SPT (Rule 4)	11456.2	12167.2	1900.2
SSPT (Rule 9)	10731.9	11529.1	1882.85
Difference	724.3	638.1	17.35
% Difference	6.32%	5.24%	0.91%
STPT (Rule 5)	9223.15	9902	1878.35
SSTPT (Rule 10)	8607.71	9338.36	1782.5
Difference	615.44	563.64	95.85
% Difference	6.67%	5.69%	5.10%

From Table 8, the difference between the modified priority rules with setup times consideration and the classical priority rules without setup times consideration based on total progressive weighted tardiness can be arranged from the greatest value of the difference to the smallest value of the difference as follows:

SMST rule is different from SMT rule by 23.26%,
MWKRS rule is different from MWKR rule by 7.42%,
LWKRS rule is different from LWKR rule by 7.07%,
SSTPT rule is different from STPT rule by 6.67%, and
SSPT rule is different from SPT rule by 6.32%.
The average of the difference is 9.28%.

Based on the total earliness and progressive weighted tardiness, the difference between the modified priority rules with setup times consideration and the classical priority rules without setup times consideration can be arranged from the greatest value of the difference to the smallest value of the difference as follows:

SMST rule is different from SMT rule by 21.84%,
MWKRS rule is different from MWKR rule by 6.90%,
LWKRS rule is different from LWKR rule by 6.72%,
SSTPT rule is different from STPT rule by 5.69%, and
SSPT rule is different from SPT rule by 5.24%.
The average of the difference is 10.15%.

Based on the total earliness and tardiness, the difference between the modified priority rules with setup times consideration and the classical priority rules without setup times consideration can be arranged from the greatest value of the difference to the smallest value of the difference as follows:

LWKRS rule is different from LWKR rule by 9.70%,
SMST rule is different from SMT rule by 8.56%,
SSTPT rule is different from STPT rule by 5.10%,
MWKRS rule is different from MWKR rule by 1.88%,
and SSPT rule is different from SPT rule by 0.91%.
The average of the difference is 5.232%.

It can be concluded that, based on the following measures of performance: (i) total progressive weighted tardiness (ii) earliness and progressive weighted tardiness and (iii) total earliness and tardiness, the modified priority rules with sequence-dependent setup times are superior to the classical priority rules.

8. THE PROPOSED MPWT HEURISTIC METHOD

In this paper, a new heuristic approach to solve the conflicting operations in the set of active and nondelay

schedules is proposed and compared with other efficient heuristics. This new heuristic method with sequence-dependent setup times consideration is developed to solve the new job shop measures of performance as follows: (i) total earliness and progressive weighted tardiness and (ii) total progressive weighted tardiness. The objective of Mean Progressive Weighted Tardiness Estimator (MPWT) heuristic method is to determine the estimate of the job (minimum transport batch) completion time, and then to estimate the mean progressive weighted tardiness. Therefore, it is named "Mean Progressive Weighted Tardiness Estimator (MPWT) Heuristic Method". The flow chart of the MPWT heuristic procedure is displayed in Figure 6.

The calculation steps of the MPWT heuristic method are as follows.

Step 1 Set the initial value – *MinMPWT* = the maximum value.

Step 2 Do the following loop until all conflicting operations have been considered. If the number of loop variable I < the number of conflicting operations in the set of active and nondelay schedules, then follow the next step; otherwise go to Step 13.

Step 3 Set the initial values – $MPWT = 0$ and $N = 0$.

Step 4 Repeat the loop in every job and assume that each conflicting operation is added in the partial sche-dule, then go to Step 8.

Step 5 Calculate the start time of operation I. Then find the machine that this operation I works on, and find the finish time of the remaining operation I based on work remaining of the considered job (the estimate of job completion time).

Step 6 Compare the estimate of job completion time and the job due date. If the estimate of job completion time is greater than the job due date (the job is estimated to be late), set N (the estimate value of the number of tardy jobs) = $N + 1$, *Progressive Tardiness* = (*estimate of job completion time – due date*) * *Penalty Weight* * (1+ *progressive rate*)$^{N-1}$, *TotalProgressiveWeighted Tardiness* = *TotalProgressiveWeightedTardiness* + *Progressive Tardiness*.

Step 7 Consider the next job, and then go to Step 4.

Step 8 Calculate the estimate of mean progressive weighted tardiness by calculating *MPWT* = *TotalProgressiveWeightedTardiness* / *Number of total jobs*.

Step 9 Set data to the current job and machine data.

Step 10 Find the minimum value of *MPWT*.

Step 11 Shift to the next job, consider the next job data, and then go back to Step 3.

Step 12 Select the conflicting operation whose job has the minimum *MPWT*.

Step 13 Determine the appropriate sequence-dependent setup times to be added in the value of variables for calculation in the next iteration, and consider the next operation.

To illustrate the numerical example of the proposed MPWT heuristic method, consider the numerical example shown in Section 6.

The MPWT heuristic approach can be applied in the set of active schedules and nondelay schedules. However, it would be better to solve the conflicting operations in the set of active schedules, since the set of nondelay schedules is the subset of active schedules. Therefore, the numerical example of modified active schedule generation algorithm using the MPWT heuristic method to solve the conflicting operations in the set of active schedules is shown below.

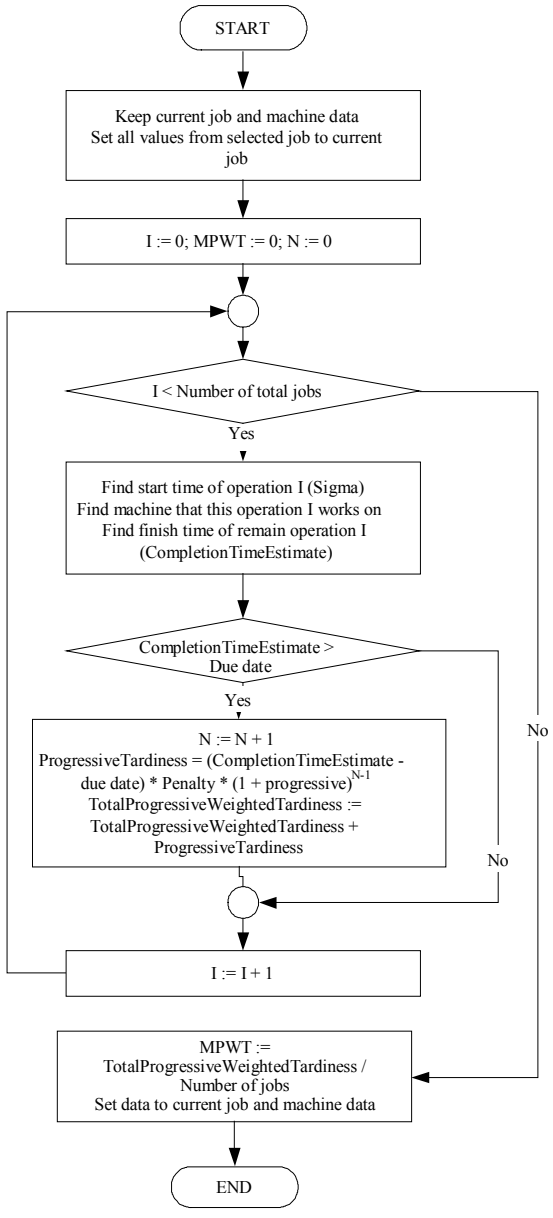


Figure 6. Flow chart of the MPWT heuristic method

As in Section 6, the beginning of the complete schedule is 01 February 2004 8:00 a.m., and the scheduling data that are job routing, processing times, and inter-machine overlapping sequence dependent setup times are the same.

Loop : 1
PS : {Empty Set}
Phi (the earliest finish time of each operation):
(1,1,3 := 01-Feb-04 08:00:00+0+4
= 01-Feb-04 08:04:00),
(2,1,2 := 01-Feb-04 08:00:00+0+3
= 01-Feb-04 08:03:00),
(3,1,1 := 01-Feb-04 08:00:00+0+3
= 01-Feb-04 08:03:00)
Phi* (the smallest value of Phi) :
01-Feb-04 08:03:00
M* : 2,1
Active Schedulable Operations & Estimated MPWT:
(2,1,2) :=
JOB 1 => Estimated Completion Time
= Min((1,1,3) := 01-Feb-04 08:00:00+0+33
= 01-Feb-04 08:33:00) = 01-Feb-04 08:33:00;
Due Date = 01-Feb-04 09:34:00;
JOB 2 => Estimated Completion Time
= Min((2,2,3) := 01-Feb-04 08:03:00+0+32
=01-Feb-04 08:35:00) = 01-Feb-04 08:35:00;
Due Date = 01-Feb-04 09:38:00;
JOB 3 => Estimated Completion Time
= Min((3,1,1) := 01-Feb-04 08:00:00+0+18
= 01-Feb-04 08:18:00) = Feb-04 08:18:00;
Due Date = 01-Feb-04 10:15:00;
Estimated Mean Progressive Weighted Tardiness = 0
No. of Tardy Job = 0
(3,1,1) := JOB 1=> Estimated Completion Time
= Min((1,1,3) := 01-Feb-04 08:00:00+0+33
= 01-Feb-04 08:33:00) = 01-Feb-04 08:33:00;
Due Date = 01-Feb-04 09:34:00;
JOB 2=> Estimated Completion Time
= Min((2,1,2) := 01-Feb-04 08:00:00+0+35
= 01-Feb-04 08:35:00) = 01-Feb-04 08:35:00;
Due Date = 01-Feb-04 09:38:00;
JOB 3=> Estimated Completion Time
= Min((3,2,2) := 01- Feb-04 08:03:00+0+15
=01-Feb-04 08:18:00)=01-Feb-04 08:18:00;
Due Date = 01-Feb-04 10:15:00;
Estimated Mean Progressive Weighted Tardiness = 0
No. of Tardy Job = 0
Selected Job : (2,1,2=0)

Loop : 2
PS : {(2,1,2)}
Phi : (1,1,3 := 01-Feb-04 08:00:00+0+4
= 01-Feb-04 08:04:00),
(2,2,3 := 01-Feb-04 08:03:00+0+30
= 01-Feb-04 08:33:00),
(3,1,1 := 01-Feb-04 08:00:00+0+3
= 01-Feb-04 08:03:00)
Phi* : 01-Feb-04 08:03:00
M* : 1

(3,1,1)
Active Schedulable Operations & Estimated MPWT:
JOB 1=> Estimated Completion Time
= Min((1,1,3) := 01-Feb-04 08:00:00+0+33
= 01-Feb-04 08:33:00) = 01-Feb-04 08:33:00;
Due Date = 01-Feb-04 09:34:00;
JOB 2 => Estimated Completion Time
= Min((2,2,3) := 01-Feb-04 08:03:00+0+32
= 01-Feb-04 08:35:00) = 01-Feb-04 08:35:00;
Due Date = 01-Feb-04 09:38:00;
JOB 3 => Estimated Completion Time
= Min((3,2,2) := 01-Feb-04 08:03:00+57+15
= 01-Feb-04 09:15:00) = 01-Feb-04 09:15:00;
Due Date = 01-Feb-04 10:15:00
Estimated Mean Progressive Weighted Tardiness = 0
No. of Tardy Job = 0
Selected Job : (3,1,1=0)

Loop : 3
PS : {(2,1,2),(3,1,1)}
Phi : (1,1,3 := 01-Feb-04 08:00:00+0+4
=01-Feb-04 08:04:00),
(2,2,3 := 01-Feb-04 08:03:00+0+30
= 01-Feb-04 08:33:00),
(3,2,2 := 01-Feb-04 08:03:00+57+12
=01-Feb-04 09:12:00)
Phi* : 01-Feb-04 08:04:00
M* : 3
Active Schedulable Operations & Estimated MPWT:
(1,1,3) :=
JOB 1=> Estimated Completion Time
= Min((1,2,2) := 01-Feb-04 08:03:00+26+29
=01-Feb-04 08:58:00)= 01-Feb-04 08:58:00;
Due Date = 01-Feb-04 09:34:00;
JOB 2=> Estimated Completion Time
= Min((2,2,3) := 01-Feb-04 08:04:00+36+32
= 01-Feb-04 09:12:00) = 01- Feb-04 09:12:00;
Due Date = 01-Feb-04 09:38:00;
JOB 3 => Estimated Completion Time
= Min((3,2,2) := 01-Feb-04 08:03:00+57+15
= 01-Feb-04 09:15:00) = 01-Feb-04 09:15:00;
Due Date = 01-Feb-04 10:15:00;
Estimated Mean Progressive Weighted Tardiness = 0
No. of Tardy Job = 0
(2,2,3) :=
JOB 1 => Estimated Completion Time
= Min((1,1,3) := 01-Feb-04 08:33:00+43+33
= 01-Feb-04 09:49:00) = 01-Feb-04 09:49:00;
Due Date = 01-Feb-04 09:34:00;
JOB 2=> Estimated Completion Time
= Min((2,3,1) := 01-Feb-04 08:20:00+13+2
=01-Feb-04 08:35:00) = 01-Feb-04 08:35:00;
Due Date = 01-Feb-04 09:38:00;

JOB 3=> Estimated Completion Time
= Min((3,2,2) := 01-Feb-04 08:03:00+57+15
= 01-Feb-04 09:15:00)= 01-Feb-04 09:15:00;
Due Date = 01-Feb-04 10:15:00;
Estimated Mean Progressive Weighted Tardiness = 5
No. of Tardy Job = 1
Selected Job : (1,1,3=0)
Do loop from loop 4 to loop 9 until a complete schedule is obtained.

Table 9. The results of the numerical example

Heuristics	Total Earliness & Progressive Weighted Tardiness	Total Earliness & Tardiness
MPWT	88	52
MWKR	358	106
SSPT, SSTPT	631	99
SMST	631	99
LWKR	631	99

From Table 9, total earliness and progressive weighted tardiness, and total earliness and tardiness of the complete active schedule obtained by the MPWT heuristic method are smaller than that of other modified priority rules.

9. MPWT METHOD EXPERIMENTAL RESULTS AND ANALYSIS

Firstly, the scheduling algorithms to generate active or nondelay schedules are selected. Secondly, in the set of active or nondelay schedules, the proposed MPWT heuristic method is used to solve the conflicting operations as shown in Figure 7.

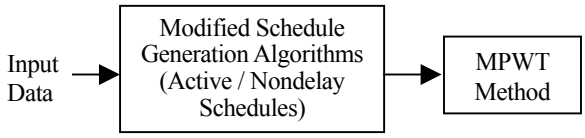


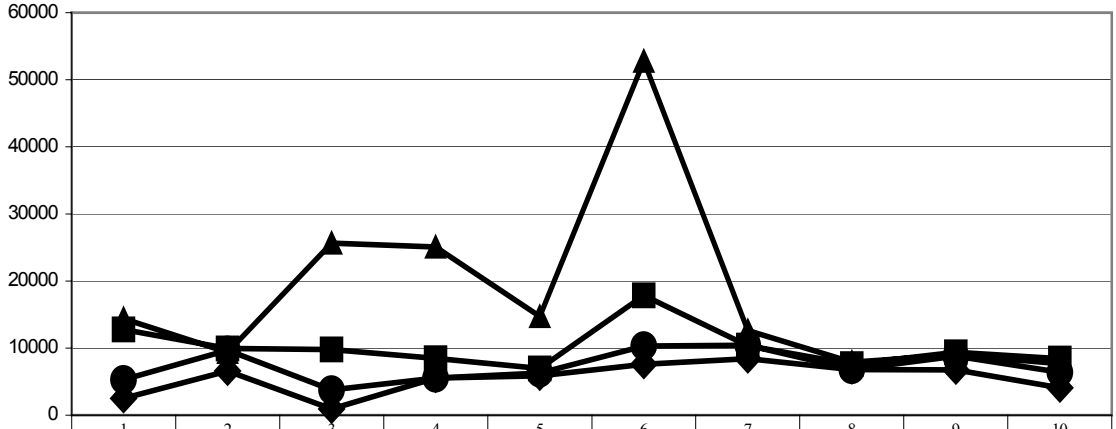
Figure 7. Flow diagram of the MPWT heuristic method

In order to compare the MPWT heuristic approach with other efficient heuristics, the efficient heuristics that are selected to use in the second experiment are as follows.

- Batched Apparent Tardiness Cost with Sequence-dependent Setups (BATCS) rule
- LWKRS rule
- SMST rule

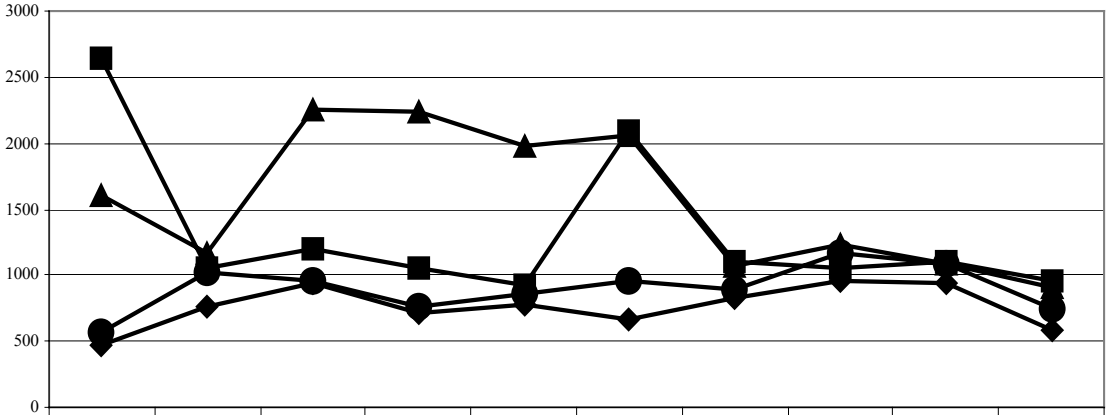
From Figure 8, it is obvious that based on the total earliness and progressive weighted tardiness, our proposed heuristic (MPWT method) and the modified priority rule (SMST) are superior to the BATCS (Batched Apparent Tardiness Cost with Sequence-dependent Setups) rule. The BATCS rule has been modified from the Batched Apparent Tardiness Cost (BATC) recently presented in Balasubramanian *et al*. (2004). The generalization of the Apparent Tardiness Cost (ATC) has been developed in many papers (see Pinedo 1995, Jeong and Kim 1998, Jayamohan and Rajendran 2000, Thiagarajan and Rajendran 2003, and Balasubramanian *et al*. 2004). Based on the total earliness and progressive weighted tardiness, the MPWT heuristic method is superior to other heuristics.

According to Figure 9, the result obtained by the experiment based on total earliness and tardiness shows that the MPWT approach is superior to the SMST, BATCS, and LWKRS rules.



	1	2	3	4	5	6	7	8	9	10
LWKRS	14391.43	9358.3	25656.77	25075.02	14696.04	52790.77	12639.68	7913.78	9030.75	7693.46
BATCS	12760.11	9944.96	9765.97	8487.55	6938.81	17872.1	10353.38	7592.94	9377.51	8409.3
SMST	5275.89	9677.07	3757	5562.84	6276	10336	10407.01	6781.56	8800.31	6379.12
MPWT	2495.2	6576.1	964	5562.84	5869.79	7614.83	8408.5	6781.56	6751.2	4073.26

Figure 8. MPWT experimental results based on total earliness and progressive weighted tardiness



	1	2	3	4	5	6	7	8	9	10
LWKRS	1611	1173	2252	2234	1980	2058	1073	1237	1094	902
BATCS	2648	1053	1194	1047	919	2094	1102	1057	1097	953
SMST	566	1027	964	765	862	958	890	1170	1090	747
MPWT	470	759	937	715	775	662	826	955	939	589

Figure 9. MPWT experimental results based on total earliness and tardiness

The next section is the result of the implementation of the proposed MPWT heuristic method in the case study. An automotive parts factory is selected to be a case study. Since the job shop scheduling algorithms are too complicated to calculate manually, the proposed sequential heuristic approach is coded in the interactive scheduling software designed and developed by using well known Visual Programming Languages (VPL), i.e. Visual C++ and Visual Basic. Its objective is to demonstrate the effecttiveness of the proposed scheduling algorithm implemented in the automotive parts factory, which is a job shop environment. The effectiveness of the implementation of the proposed sequential heuristic method, with the consideration of more realistic constraints, will be investigated in the job shop production system in the next section.

10. COMPARISON BETWEEN THE MPWT HEURISTIC METHOD AND OTHER EFFICIENT HEURISTICS BASED ON THE SCHEDULING DATA OF THE CASE STUDY

In the comparison between the proposed MPWT heuristic method and other efficient heuristics in this section, scheduling data are collected from the production order and manufacturing process form and purchasing order form. Since the case-study factory production environment is a make-to-order production system, each step of production planning depends on data in the purchasing order form. These data are in the factory document called "delivery plan".

The delivery plan includes customer names, part numbers, part names, purchasing order numbers, due dates, part quantity and job quantity needed by customers, actual delivered job quantity, difference between job quantity and actual delivered job quantity, and difference between part quantity and actual delivered job quantity.

Additionally, the other scheduling data are in the production order and manufacturing process form. The production order and manufacturing process form includes production order number, part number, part name, job quantity needed by customer due date, detail production process, machine, setup time, processing time, total processing time, and order date. Job data, operation data, and other details of the job are used in the developed scheduling software. In order to make the comparison correct and conforming with the real production environment, there are production constraints in the scheduling software. For example, resource calendar, inter-machine overlapping sequence-dependent setup times are taken into account. Usually there are three types of working shifts, depending upon the workload of each machine, as follows: one shift, two shifts, and three shifts. The machines, which are often operated 24-hour and used with three working shifts, are band saws and circular saw. The machines that work two shifts are CNC two-spindle lathe, CNC turret lathe, etc. Usually the machines that work one shift are tapping machines, drills, milling machines, press, manual lathes, etc. If there are tardiness problems, machine working time is expanded in the overtime.

The full factorial design with ten sets of the real scheduling data (replications) are used in the comparison. The analysis of variance (ANOVA) is summarized in Table 9. The 5% significance level is used in this experiment. The P-value tests the statistical significance of each factor. According to the main effect, the P-value for the

Table 10. Analysis of variance for total earliness and progressive weighted tardiness

Factor	Type	Levels	Values
Scheduling Algorithms	fixed	2	1 (Active Schedules) 2 (Nondelay Schedules)
Heuristics	fixed	15	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Analysis of Variance using Adjusted Sum of Square for Tests

Source	Sum of Square	Df	Mean Square	F-Ratio	P-Value
Main Effects					
A : Scheduling Algorithms	684,770,221	1	684,770,221.00	19.36	0.000
B : Heuristics	863,166,876	14	61,654,776.86	1.74	0.047
Interaction					
AB	942,663,988	14	67,333,142.00	1.90	0.026
Residual	9,548,055,896	270	35,363,169.99		
Total	1.2039E+10	299			

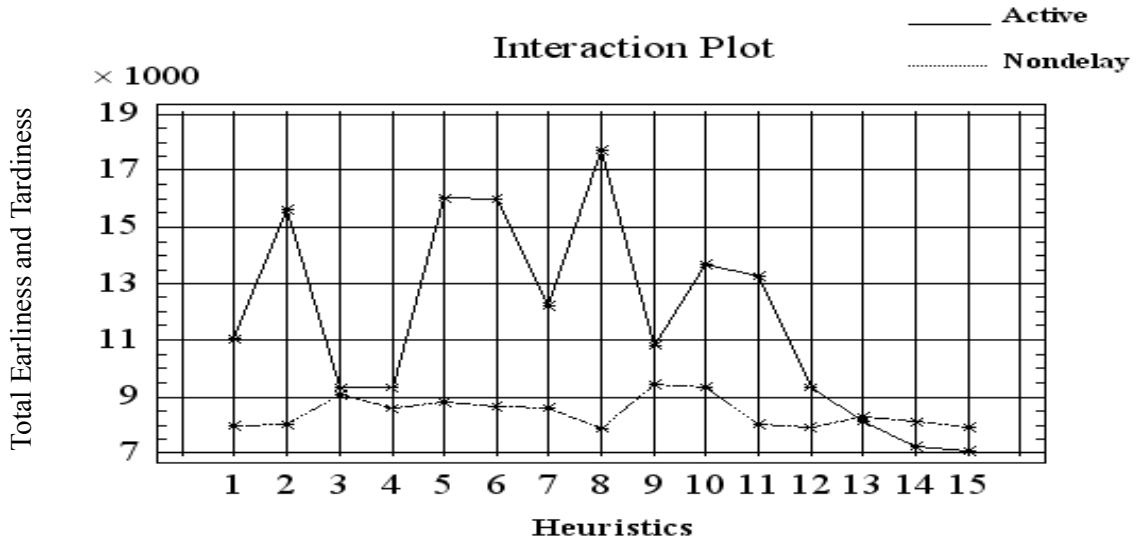


Figure 10. Interaction Plot based on total earliness and tardiness

first factor, which are the modified active and nondelay schedule generation algorithms, is lower than the significance level. Additionally, the P-value for the second factor, which are heuristics, is lower than 0.05. Therefore, it can be concluded that both factors have a statistically significant effect on total earliness and progressive weighted tardiness. The P-value for interaction between the factors is lower than the level of significance. Therefore, there is significant interaction effect based on total earliness and progressive weighted tardiness.

Figure 10. displays the plot of interactions based on the total earliness and tardiness.

The experimented scheduling generation algorithms are as follows.

Algorithm 1 : Modified active schedule generation algorithm
Algorithm 2 : Modified nondelay schedule generation algorithm

The experimented heuristics are as follows.

Heuristic 1 : EDD
Heuristic 2 : LWKR
Heuristic 3 : MWKR
Heuristic 4 : MOPNR
Heuristic 5 : SMT
Heuristic 6 : SPT
Heuristic 7 : STPT
Heuristic 8 : LWKRS
Heuristic 9 : MWKRS
Heuristic 10 : SSTPT
Heuristic 11 : SSPT
Heuristic 12 : SMST
Heuristic 13 : BATCS
Heuristic 14 : Sequential N_T-T-$E\&T$ heuristic approach
Heuristic 15 : MPWT heuristic method

The sequential N_T-T-$E\&T$ heuristic approach was recently proposed by Mongkalig (2005). It can be concluded that schedule generation algorithms and heuristics generating schedules with the smallest total earliness and tardiness in the first three approaches are: (i) modified active schedule generation algorithm using the MPWT heuristic method; (ii) modified active schedule generation algorithm using the sequential N_T-T-$E\&T$ heuristic approach; and (iii) modified nondelay schedule using the SMST priority rule. It can be concluded that the modified

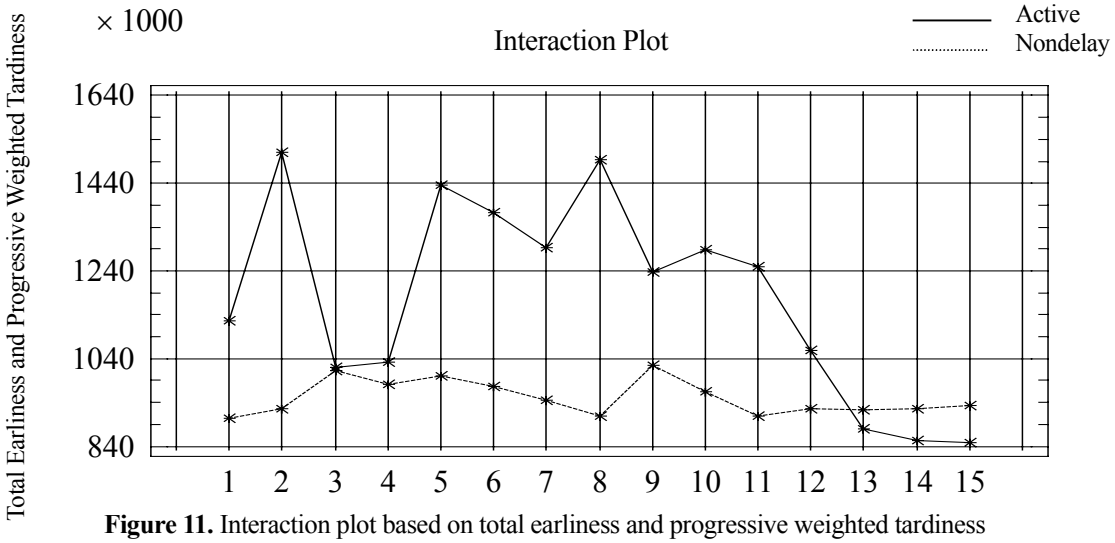


Figure 11. Interaction plot based on total earliness and progressive weighted tardiness

active schedule using the MPWT heuristic method is superior to other heuristics based on total earliness and tardiness measure of performance.

Figure 11 shows the plot of interactions based on the total earliness and progressive weighted tardiness. Schedule generation algorithms and heuristics generating schedules with the smallest total earliness and progressive weighted tardiness in the first three approaches are: (i) modified active schedule generation algorithm using the MPWT heuristic method; (ii) modified active schedule generation algorithm using the sequential N_T-*T*-*E*&*T* heuristic approach; and (iii) modified active schedule using the BATCS rule. It can be concluded that the modified active schedule generation algorithm using the MPWT heuristic method is superior to other heuristics based on total earliness and progressive weighted tardiness. Obviously, the proposed MPWT heuristic method is significantly superior to the BATCS, SMST, and other efficient priority rules based on: (i) total earliness and progressive weighted tardiness; and (ii) total earliness and tardiness.

11. CONCLUSIONS

In this paper, the MPWT heuristic method and new five priority rules, LWKRS, MWKRS, SMST, SSPT, and SSTPT, are proposed and used to compare with the efficient rules, which are the BATCS rule and the priority rules. There are three important experiments in the research. The objective of the first experiment is to determine the necessity of sequence-dependent setup time consideration when we consider the situations of job shop scheduling problems with sequence-dependent setup times, such as in the plastics, chemical, and automotive parts industries. The job shop scheduling problems with sequence-dependent setup times considered in this paper are more complex, since the realistic inter-machine overlapping setup times are taken into account. The inter-machine overlapping sequence-dependent setup can reduce the waiting time of a succeeding operation since its setup can be initiated immediately after only a fraction of the batch parts has completed the preceding operation. The reason is that there are some parts (work pieces) that have completed the preceding operation, and they are available to use in the setup of the succeeding operation. However, the important constraints of the new job shop problems with inter-machine overlapping sequence-dependent setup times are as follows:

(i) the beginning of setup time of the succeeding operation cannot occur before the completion of machine setup of the preceding operation since there is no part that can be used in the setup of the succeeding operation;
(ii) in realistic production environments, although the setup time can be overlapped; precedence constraints of the direct processing times exist. In other words, the direct processing time of the succeeding operation cannot be overlapped, and it should be started not earlier than the completion time of the preceding operation.

In addition, the contributions of this paper are not only the new MPWT heuristic method and new modified priority rules, but also the new inter-machine overlapping sequence-dependent setup constraints, and the new scheduling performance measures as follows: (i) total progresssive weighted tardiness; and (ii) total earliness and progressive weighted tardiness. The new scheduling measures of performance based on progressive weighted tardiness penalties are developed and used as the criteria of the experiments. Currently, satisfaction of customers is

through the quality of products and service. The new measures of performance have to be solved in a realistic production system when repeated tardiness of the same customer order occurs. The total progressive weighted tardiness of schedules obtained by the modified priority rules with setup times consideration – LWKRS, MWKRS, SSPT, SMST, and SSTPT rules is lower than the total progressive weighted tardiness of schedules obtained by the classical priority rules. Compared with the classical standard priority rules, the average total progressive weighted tardiness of the schedules obtained by the modified priority rules with sequence-dependent setup times consideration decreases by 6.88%. In addition, the average total earliness and progressive weighted tardiness of the schedules obtained by the modified priority rules with sequence-dependent setup times consideration decreases by 6.205%. And the average total earliness and tardiness of the schedules obtained by the modified priority rules with sequence-dependent setup times consideration decreeases by 2.303%.

The objective of the second experiment is to compare the proposed MPWT heuristic method, the BATCS rule, and the modified priority rules. The results obtained by the second experiment indicate that the proposed MPWT heuristic method is significantly superior to BATCS, SMST, and LWKRS based on total earliness and progressive weighted tardiness, and total earliness and tardiness.

The third experiment aims to compare the MPWT heuristic method with other efficient heuristics based on the scheduling data of the case study, which is a real job shop environment. From the experimental results, the scheduling generation algorithm and heuristics for solving the conflicting operations in the set of active and non-delay schedules have a statistically significant effect on total earliness and progressive weighted tardiness. There is significant interaction effect based on (i) total earliness and progressive weighted tardiness; and (ii) total earliness and tardiness. The experimental results indicate that the MPWT heuristic method is superior to the BATCS, SMST, and other efficient heuristics based on both progressive weighted tardiness penalties.

In future research, the new job shop scheduling problem should be considered. In particular, new measures of performance need to be developed which focus on customer satisfaction, because this will give the manufacturers competitive advantage.

REFERENCES

Balasubramanian, H., Mönch, L., Fowler, J. W. and Pfund, M. E. (2004), Genetic Algorithms based scheduling of parallel batch machines with incompatible job families to minimize total weighted tardiness, *International Journal of Production Research*, 42, 1621-1638.

Barman, S. (1997), Simple priority rule combinations: an approach to improve both flow time and tardiness", *International Journal of Production Research*, 35, 2857-2870.

Dearing, P. M. and Anderson, R. A. (1984), Assigning looms in a textile weaving operation with change-over limitations, *Production and Inventory Management*, 25, 23-31.

Franca, P. M., Gendreau, M., Laporte, G. and Muller, F. M. (1996), A tabu search heuristic for the multiprocessor scheduling problem with sequence dependent setup times, *International Journal of Production Economics*, 43, 79-89.

Giffler, B. and Thompson, G. L. (1960), Algorithms for solving production-scheduling problems, *Operations Research*, 8, 487-503.

Jayamohan, M. S. and Rajendran, C. (2000), New dispatching rules for shop scheduling: a step forward, *International Journal of Production Research*, 38, 563-586.

Jeong, K. C. and Kim, Y. D. (1998), A real-time scheduling mechanism for a flwexible manufacturing system: using simulation and dispatching rules, *International Journal of Production Research*, 36, 2609-2626.

Kurz, M. E. and Askin R. G. (2001), Heuristic scheduling of parallel machines with sequence-dependent set-up times, *International Journal of Production Research*, 39, 3747-3769.

Mongkalig, C. (2005), Heuristics for Job Shop Scheduling Problems with Progressive Weighted Tardiness Penalties and Inter-machine Overlapping Sequence-dependent Setup Time, Dissertation.

Monma, C. and Potts, C. N. (1989), On the complexity of scheduling with batch setup times, *Operations Research*, 37, 798-804.

Picard, J. C. and Queyranne, M. (1978), "The time dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling", *Operations Research*, 26, 86-110.

Pinedo, M. (1995), *Scheduling Theory, Algorithms, and Systems*, Prentice-Hall, Englewood Cliff, NJ.

Rajendran, C. and Holthaus, O. (1999), A comparative study of dispatching rules in dynamic flow shops and job shops, *European Journal of Operational Research*, 116, 156-170.

Reeja, M. K. and Rajendran, C. (2000), Dispatching rules for scheduling in assembly job Shops, *International Journal of Production Research*, 38, 2349-2360.

Sumichrast, R. and Baker, K. R. (1987), Scheduling parallel processors: an integer linear programming based heuristic for minimizing setup time, *International Journal of Production Research*, 25, 761-771.

Thiagarajan, S. and Rajendran, C. (2003), Scheduling in dynamic assembly job-shops with jobs having different holding and tardiness costs, *International Journal of Production Research*, 41, 4453-4486.

Uzsoy, R., Martin-Vega, L. A., Lee, C. Y. and Leonard, P. A. (1991), Production algorithms for semiconductor test facility, *IEEE Transactions on Semiconductor Manufacturing*, 4, 270-280.

Uzsoy, R., Lee, C. Y. and Martin-Vega, L. A. (1992), Scheduling semiconductor test operations: minimizing maximum lateness and number of tardy jobs on a single machine, *Naval Research Logistics*, 39, 369-388.

Vepsalainen, A. P. J. and Morton, T. E. (1987), Priority rules for job shops with weighted tardiness costs, *Management Science*, 33, 1035-1047.