Proceedings of the 2012 International Conference on Industrial Engineering and Operations Management
Istanbul, Turkey, July 3 – 6, 2012

Minimizing the Expected Number of Tardy Jobs in Dynamic Flow Shop with Stochastic Due Dates

Ali Elyasi and Nasser Salmasi
Department of Industrial Engineering
Sharif University of Technology, Tehran, Iran

Abstract

The problem of minimizing the expected number of tardy jobs in a dynamic m machine flow shop i.e., $F_m|r_j|\mathrm{E}[\sum U_j]$ is studied in this research. It is assumed that the jobs with deterministic processing times and stochastic due dates arrive dynamically and randomly to the flow shop cell. The due date of each job is assumed to be normally distributed with known mean and variance. A dynamic method is proposed for this problem by which the m machine stochastic flow shop problem is decomposed into m stochastic single machine sub-problems. Then, each sub-problem is solved as an independent stochastic single machine scheduling problem by a mathematical programming model. The proposed dynamic method is applied whenever a sub-problem is generated during the process. Comparison of the proposed method with Shortest Process Time (SPT) first rule, which is the most effective rule in literature for the proposed research problem, shows that the proposed method performs 23.9% better than SPT rule on average for industry-size scheduling problems.

Keywords
Stochastic scheduling, flow shop scheduling, tardy jobs, dynamic scheduling, release dates

1. Introduction
Most of the researches in the area of scheduling have focused on deterministic scheduling problems in which all parameters such as processing times, due dates, and release dates have fixed values and are known in advance. However, in most of practical situations, there are many sources of uncertainty such as machine breakdowns, change in due dates, release of unexpected jobs, and imprecise processing times. The use of deterministic models for stochastic environments could lead to poor solutions very far from the optimal solution. Thus, stochastic optimization concepts are used as an approach to address the uncertainty of parameters in scheduling area.
Stochastic problems are classified into static and dynamic models. In static models all jobs are available at the beginning of planning horizon and jobs are ordered in a priority list. Once the priority list is generated, it is unchanged during the planning horizon. On the other hand, in dynamic models the jobs arrive to the shop over time and join to the queue of the first machine. In dynamic models, the order of jobs to be processed can be revised at any time. The scheduler uses the available data by then and set the priority list without considering the jobs arrive later. Thus, the priority list can be changed during the planning horizon. Depending on whether the preemption is allowed or not, Pindeo [1] considers four types of policies in stochastic scheduling: (i) nonpreemptive static list policy, (ii) preemptive static list policy, (iii) nonpreemtive dynamic policy, and (iv) preemptive dynamic policy. In this research we address the stochastic multi-stage dynamic flow shop problem to minimize the expected number of tardy jobs in nonpreemtive dynamic policy. Based on Pinedo [1], the proposed research problem can be shown as $F_m|r_j|\mathrm{E}[\sum U_j]$.

2. Literature Review
Most of the researches for minimization of the expected number of tardy jobs have been performed in single machine problems. Moore [2] proposes a polynomial time algorithm named *Moore-Hodgson algorithm* to solve a single machine problem with minimization of the total number of tardy jobs $(1\|\sum U_j)$ optimally. The stochastic counterpart of this problem is introduced by Balut [3] and later proven to be NP-hard by Kise and Ibaraki [4]. Pinedo [5] analyzes the $1\|\mathrm{E}[\sum w_j U_j]$ in which the processing times of jobs are generated independently based on a randomly exponential distribution. The due dates are also generated randomly based on an exponential distribution with equal mean. He proposes a polynomial time algorithm named Shortest Expected Processing Time (SEPT) to

optimally solve this problem. Boxma and Forst [6] investigate several special cases of single machine and flow shop scheduling problems with minimizing the expected weighted number of tardy jobs. Emmons and Pinedo [7] describe several special cases on parallel machine problem with minimizing the expected number of tardy jobs $\left(P_m \| \mathrm{E}\left[\sum U_j\right]\right)$ and propose optimal policies. Sarin *et al.* [8] study a single machine problem $1\left|d_j = d\right|\mathrm{E}\left[\sum w_j U_j\right]$ in which the processing times of jobs are generated independently based on a normal distribution with the variances in which related to the mean of the processing times. The weight of each job is also related to the mean of the processing time. For this problem, they prove that a necessary, but not sufficient condition for a job sequence considered to be optimal is that the sequence must be W-shaped or V-shaped with respect to mean processing times. They used this property to substantially reduce the number of sequences that should be considered for a branch and bound algorithm. De *et al.* [9] study the stochastic counterpart of the single machine problem $1\left|D_j = D\right|\sum w_j U_j$ in which the processing time of jobs are random variables with arbitrary but known distributions. They assume that the jobs have a common, exponentially distributed due date. They derive sufficient conditions for the existence of a job sequence which stochastically minimize the weighted number of tardy jobs. They also propose a simple sequencing rule which stochastically minimizes the number of tardy jobs. Jang [10] investigate $1\left|r_j\right|\mathrm{E}\left[\sum U_j\right]$ when the processing time of jobs are generated based on a normal distribution. He develops a dynamic policy based on a myopically optimal solution, by which a simple and robust dynamic policy can be obtained. Seo *et al.* [11] study $1\left|d_j = d\right|\mathrm{E}\left[\sum U_j\right]$ problem in which jobs have normally distributed processing times and develop some mathematical models for this problem. Akker and Hoogeveen [12] present a model for stochastic counterpart of $1\left\|\sum w_j U_j\right.$ using a chance constraint to define whether a job with stochastic processing time is on time or late. They analyze several special cases that could be solved in polynomial time by Moore-Hodgson [2] algorithm. Trietsch and Baker [13] unify and generalize the Akker and Hoogeveen's results for any stochastically ordered processing times. The authors present a polynomial time algorithm to stochastically minimize the number of tardy jobs.

To the best of our knowledge, the only available research that considers minimizing the expected number of tardy jobs in stochastic flow shop is the study performed by Boxma and Forst [9] in which two special cases are investigated. In both cases, all due dates are assumed to be independent and identically distributed. So, there are no results for flow shop problems with non-identically distributed due dates. Based on real world application of stochastic flow shop scheduling problems, there are rooms to perform more research in this area.

In this research, a dynamic method for stochastic flow shop scheduling is proposed to minimize the expected number of tardy jobs. The proposed method is based on decomposition of *m* machine stochastic flow shop problem into m stochastic single machine sub-problems. Then, each sub-problem is solved as an independent stochastic single machine scheduling problem by a mathematical programming model. The proposed dynamic method is applied whenever a sub-problem is generated during the process.

3. Problem Description

Consider a flow shop cell including m machines which n jobs should be processed on these machines. The other specifications of the proposed research problem are as follows:

- Each job has a release date $\left(r_j\right)$ which is not determined at the beginning of planning horizon. The scheduler will consider each job in the planning as soon as the job arrives to the flow shop cell.
- Each job has a stochastic due date $\left(D_j\right)$ which is a random variable with normal distribution with determined mean $\left(\mu_j\right)$ and variance $\left(\sigma_j^2\right)$.
- The processing time of each job on each machine $\left(p_{ij}\right)$ has a deterministic value and is provided once the job arrives to the flow shop cell.
- All jobs have equal weights. In other words, the jobs have equal importance.
- The objective function is to minimize the value of the expected number of tardy jobs $\left(\mathrm{E}\left[\sum U_j\right]\right)$.
- The preemption is not allowed. In other words, if the process of a job is started on a machine, the process cannot be interrupted before completing the process.
- The sequence of processing jobs on machines can be different. In other words, the permutation assumption is not considered in this research.
- The buffer capacity between two consecutive machines is unlimited.
- It is assumed that setup time is included inside the processing times of jobs on machines.
- Machines are always available and no breakdown is occurred.

- Only non-delay schedules are considered in this study. In other words, no machine is kept idle while a job is in its queue waiting for processing.

Based on Pinedo [1], the proposed research problem can be described as $F_m|r_j|\mathrm{E}[\sum U_j]$; flow shop scheduling problem (F_m), unavailability of all jobs at the beginning of planning horizon (r_j), and minimization of expected number of tardy jobs as the criterion $(E[\sum U_j])$.
Kise and Ibaraki [4] prove that $1\|\mathrm{E}(\sum U_j)$ is an NP-hard problem. Since the research problem can be reduced to$1\|\mathrm{E}(\sum U_j)$, it can be concluded that our research problem i.e., $F_m|r_j|\mathrm{E}[\sum U_j]$ is NP-hard, respectively.

4. Solution Method

Although there is no research carried out in $F_m|r_j|\mathrm{E}[\sum U_j]$, several studies focused on this issue in deterministic case. For instance, Lodree *et al.* [14] present a rule which named EADD (Earliest Adjusted Due Date) for minimizing the number of tardy jobs in a multi stage dynamic flow shop. They assume that the release date, processing time, and due date of each job are not known in advance and once a job arrives, the value of its parameters are known. The idea in their approach is to decompose the m machine flow shop problem into m independent single machine sub-problems. Then, each sub-problem is solved optimally by using a variation of Moore-Hodgson's [2] algorithm. When the process of a job is finished on a machine, if more than one job are staying in line to be processed by the machine, a sub-problem is generated and solved in order to find the job processed as the next job on the machine in order to optimize the objective function. If only one job is in the line, there is no need to generate the sub-problem. The EADD rule is applied whenever a sub-problem is generated during the process. Lodree *et al.* [14] assume that the due dates of jobs are deterministic and have a non-negative integer value. Also, they assume that the due date of a job is known once it arrives. In order to locally solve any single machine sub-problem, they adjust the due date of jobs to represent the job's due date on current machine for the local single machine sub-problem. The adjusted due date on a single machine sub-problem is interpreted as the time that a job has to be completed on current machine in order to prevent any delay for the job on the last machine of the flow shop cell by considering the actual due date of the job. In order to compute the adjusted due date of a job in a single machine sub-problem, the summation of the remaining processing times and summation of remaining waiting times of the job on downstream machines is subtracted from the remaining available time until the actual due date of the job. They indicate that EADD performs 15% to 20% better than the shortest processing time (SPT) rule, which was the most effective method for minimizing the number of tardy jobs in dynamic flow shop by then.
We modify the approach proposed by Lodree *et al.* [14] to solve the stochastic version of the problem. In this research, in spite of Lodree *et al.* [14], it is assumed that the jobs have stochastic due dates. Hence, the approach proposed by Lodree *et al.* [14] cannot be applied directly for stochastic version of the problem. Thus, we modify their approach to solve the proposed research problem.
In this research, the m machine stochastic flow shop problem is decomposed into m stochastic single machine sub-problems. Then, each sub-problem is solved as an independent single machine stochastic scheduling problem by a mathematical programming model. When the process of a job is finished on a machine, if more than one job are staying in line of the machine (to be processed by the machine), the sub-problem is generated and solved in order to find the job processed as the next job on the machine. Since only non-delay schedules are considered, unforced idleness of machines is prohibited. Therefore, if only one job is in a machine's queue when the machine becomes available, obviously that job has to be processed on the machine and the mathematical programming model does not need to be solved. The proposed method is applied whenever a sub-problem is generated during the process. The notations used in this research are as the followings:

t	The notation used for time (the time at which the dispatching decision is made)
i	The notation used for machines
j	The notation used for jobs
p_{ij}	The processing time of job j on machine i
D_j	The random due date of job j which is generated based on $N(\mu_j, \sigma_j^2)$
D'_{ijt}	The adjusted due date of job j on machine i at time t
q_{ij}	The duration of time that job j waits in the line of machine i
Q_{it}	The set of jobs in the queue of machine i at time t

In order to locally solve the single machine sub-problems, the due dates of jobs have to be adjusted to represent the due dates for the local single machine problems at current time. For this purpose, a similar approach proposed by Lodree *et al.* [14] is applied here. The explanation of adjusted due date of a job on a machine is the same as used in Lodree *et al.* [14]. However, as the original due dates are stochastic in this study; the due dates of jobs in single machine sub-problems will be stochastic too. Hence, in order to transform the actual due date of a job to adjusted due date of the job on a machine at any time, the summation of the remaining processing times and remaining waiting times of the job on downstream machines is subtracted from the remaining available time until the actual due date of the job. So if the scheduler makes a decision at time $t > 0$, then the expression $D_j - t$ represents the remaining available time for job *j* to meet its original due date. Consequently, the adjusted due date of job *j* on machine i at time t is given by:

$$D'_{ijt} = D_j - t - \sum_{k=i+1}^{m} q_{kj} - \sum_{k=i+1}^{m} p_{kj} = D_j - (t + \sum_{k=i+1}^{m} q_{kj} + \sum_{k=i+1}^{m} p_{kj}) \quad (1)$$

Recall that $D_j \sim N(\mu_j, \sigma_j^2)$. The expression $t + \sum_{k=i+1}^{m} q_{kj} + \sum_{k=i+1}^{m} p_{kj}$ in Eq. (1) is a constant for any specified job such as *j* at any specified time *t*. This constant value is referred to as a_{ijt} in this research. Suppose a single machine sub-problem is generated on machine i at time *t*. So, for all jobs which are in the queue of machine i at time *t*, i.e., for $\forall j \in Q_{it}$, the values of a_{ijt} can be calculated. The generated sub-problem is a single machine problem with deterministic processing times and stochastic due dates. Thus, in any single machine sub-problem, the jobs in the queue of machine i at time t have random due dates with normal distribution with mean $\mu_j - a_{ijt}$ and variance σ_j^2, i.e. $\forall j \in Q_{it}$: $D'_{ijt} \sim N(\mu_j - a_{ijt}, \sigma_j^2)$. By replacing $\mu_j - a_{ijt}$ with μ_{ijt} the distribution of adjusted due dates can be represented as $D'_{ijt} \sim N(\mu_{ijt}, \sigma_j^2)$. Consequently, the variance of due dates in sub-problems remain unchanged. Thus, in order to solve the single machine sub-problems it is only required to compute the means of adjusted due dates, i.e. μ_{ijt}. Hence, it follows that:

$$\mu_{ijt} = \mu_j - a_{ijt} = \mu_j - t - \sum_{k=i+1}^{m} q_{kj} - \sum_{k=i+1}^{m} p_{kj} \quad (2)$$

A difficulty arises during the calculation of μ_{ijt} in Eq. (2). This difficulty is due to existence of $\sum_{k=i+1}^{m} q_{kj}$ in the equation. Waiting times of each job on downstream machines depend on scheduling of all jobs in downstream machines in the flow shop. Therefore, due to the nature of dynamic scheduling, we cannot compute $\sum_{k=i+1}^{m} q_{kj}$ at time *t*. So, we should estimate μ_{ijt} with estimation of q_{ij} with following alternative equation:

$$\hat{\mu}_{ijt} = \mu_j - t - \sum_{k=i+1}^{m} \hat{q}_{kj} - \sum_{k=i+1}^{m} p_{kj} \quad (3)$$

In Eq. (3), the $\hat{\mu}_{ijt}$ and $\hat{q}_{ij}$ are estimation of μ_{ijt} and q_{ij}, respectively. We use a recursive process for estimating q_{ij}. For each job (say job *j*) belongs to Q_{it}, it is assumed that job *j* is the next job is assigned to be processed by machine i. In this case, the scheduling of the job on machine i and the downstream machines is performed by assuming temporarily that the jobs in queues of the downstream machines are processed based on first come first serve (FCFS) discipline. By this recursive process, we can calculate the $\sum_{k=i+1}^{m} \hat{q}_{kj}$ for job *j*. The recursive process is repeated for all jobs in Q_{it}. Once μ_{ijt} is calculated for all jobs in Q_{it}, the single machine sub-problem can be solved.

Now we describe how to solve the single machine sub-problems. Seo *et al.* [11] develop a mathematical programming approach for a stochastic single machine scheduling problem to minimize the expected number of tardy jobs. They assume that jobs have normally distributed processing times and a common deterministic due date. We apply their method to solve the sub-problems in this research.

As mentioned, when process of a job is finished on a machine, if more than one job are staying in line of the machine, the sub-problem is generated and solved in order to determine the next job to be processed by the machine. The sub-problem is a single machine problem with deterministic processing times (p_{ij}) and stochastic due dates for the jobs with normal distribution with determined means (μ_{ijt}) and variances (σ_j^2). The number of jobs in each single machine sub-problem is equal to the number of jobs in the queue of the specified machine at the time that the single machine sub-problem is generated. If a single machine sub-problem is generated at time t on machine *i*, the number of jobs in such problem is demonstrated by n_{it} ($n_{it} = |Q_{it}|$).

When mathematical programming approach is used for a single machine scheduling problem, all jobs should be available at the beginning of planning horizon. So in our problem, when a single machine sub-problem is generated at time t on machine i, the jobs in the queue of machine i at time t are only considered in mathematical programming. So, the solution (sequence of jobs) for any single machine sub-problem is valid until no job join to the queue of that machine. If during the process of a job on a machine, one or more new jobs join to the queue of that machine, a new single machine sub-problem is generated after the end of processing the current job.

We develop a mathematical programming model for stochastic single machine sub-problems based on Seo *et al.* [11]. Suppose a single machine sub-problem with n_{it} jobs is generated at time t on machine *i*. Let

$S = ([1], [2], \dots, [k], \dots, [n_{it}])$ is an arbitrary sequence of the n_{it} jobs on machine i and $[k]$ indicates the job occupying the k^{th} position in that sequence. Let $C_{i[k]t}$ be the remaining time until completion time of the k^{th} job on machine i in the generated sub-problem at time *t*. Then, $C_{i[k]t}$ satisfies the following equation:

$$C_{i[k]t} = \sum_{l=1}^{k} p_{i[l]} \tag{4}$$

So the expected number of tardy jobs of a sequence S in a single machine sub-problem is expressed by the following equation:

$$E\left[\sum U_j\right] = \sum_{k=1}^{n} \Pr\left(C_{i[k]t} > D'_{i[k]t}\right) = \sum_{k=1}^{n} \Pr\left(\sum_{l=1}^{k} p_{i[l]} > D'_{i[k]t}\right) \tag{5}$$

As the due dates of jobs are random variables, this problem is very hard to solve. So, we develop a mathematical programming model based on Seo *et al.* [11] to transform the stochastic model given in Eq. (5) to equivalent non-linear deterministic problem. In their research problem, jobs have a common deterministic due date and stochastic processing times which normally distributed with determined mean and variance. We contribute their proposed model to solve our sub-problems in which the jobs have deterministic processing times and stochastic due dates which normally distributed with determined mean and variance. For this purpose, a binary variable x_{jp} is defined for the sub-problem that is generated on machine i at time t as follows: $x_{jp} = 1$ if job j is scheduled at p^{th} position in the sequence, and $x_{jp} = 0$ otherwise.

In each single machine sub-problem, each job is assigned to exactly one position and each position is assigned to exactly one job. Considering the assignment constraints, the expected number of tardy jobs of a sequence S in a single machine sub-problem is equivalent as follows:

$$E\left[\sum U_j\right] = \sum_{k=1}^{n_{it}} \left[\Pr\left(\sum_{p=1}^{k} \sum_{j=1}^{n_{it}} p_{ij} x_{jp} > \sum_{j=1}^{n_{it}} D'_{ijt} x_{jk}\right)\right] \tag{6}$$

Note that $D'_{ijt} \sim N\left(\mu_{ijt}, \sigma_j^2\right)$. So, we can develop the mathematical programming model for the single machine sub-problems as follows:

$$\min \sum_{k=1}^{n_{it}} \left[\Phi\left(\frac{\sum_{p=1}^{k} \sum_{j=1}^{n_{it}} p_{ij} x_{jp} - \sum_{j=1}^{n_{it}} \mu_{ijt} x_{jk}}{\sqrt{\sum_{j=1}^{n_{it}} \sigma_j^2 x_{jk}}}\right)\right] \tag{7}$$

Subject to

$$\sum_{p=1}^{n_{it}} x_{jp} = 1, \quad j = 1, \dots, n_{it} \tag{8}$$

$$\sum_{j=1}^{n_{it}} x_{jp} = 1, \quad p = 1, \dots, n_{it} \tag{9}$$

$$x_{jp} \in \{0,1\} \quad \forall j, p \tag{10}$$

The objective function (7) is standardized and $\Phi(\cdot)$ is the cumulative distribution function for standard normal random variable. Constraints (8) ensure that each job is assigned to exactly one position and constraints (9) ensure that each position is assigned to exactly one job. Constraints (10) indicate that all variables are either zero or one. Since the model belongs to non-linear programming problem, there is no guaranty to obtain the global optimal solution for all instances. Therefore, to get assured of the globality of the solution, the approximation of the solution of sub-problems is used by linearization of the objective function. The linearization of the objective function is performed as the same as that of Seo *et al.* [11]. Since the matrix of constraints is totally unimodular, every basic solution of the linear model will be an integer vector. Therefore, the decision variable x_{ijp} can be considered as a continuous variable. As the result, the linear programming model for the single machine sub-problems can be written as follows:

$$\min \sum_{k=1}^{n_{it}} \left(\sum_{p=1}^{k} \sum_{j=1}^{n_{it}} p_{ij} x_{jp} - \sum_{j=1}^{n_{it}} \frac{\mu_{ijt}}{\sigma_j} x_{jk}\right) \tag{11}$$

Subject to:

Constraints (8) and (9) of the original model

$$x_{jp} \geq 0 \quad \forall j, p \tag{12}$$

Briefly, whenever a sub-problem is generated during the process, we decompose the stochastic flow shop problem into multiple stochastic single machine problems and solve each stochastic single machine problem by a mathematical programming model.

It is important to note that as the due date of jobs are stochastic; we do not know the exact due date of each job. Once the due date of a job has passed, that job is considered as a tardy job and it can be ignored. If this happens during processing a job on a machine, since the preemption is not allowed, the job should be processed completely on that machine. However, processing the job on downstream machines does not have any necessity. On the other hand, if the due date of a job is received when the job is waiting in queue of a machine; the job is removed

immediately from the queue. Obviously, processing the tardy job on downstream machines is ignored. The remaining processing of tardy jobs can be resumed as arbitrary at the end of the schedule.
The process of solving the single machine sub-problems by mathematical programming model will not be a difficult problem even for the problems with a large number of jobs for two reasons. The first reason is that when processing a job on a machine is finished, it is not always necessary to solve the mathematical programming model. In two cases we do not need to solve the mathematical programming model: (i) when processing a job on a machine is finished, the number of jobs in the queue of that machine is not greater that one; (ii) when processing a job on a machine is finished, the previous solution (sequence of jobs) for this machine is still valid. If we have already solved the mathematical programming model for this machine and from then till now no job joins the queue of the machine, the previous solution (sequence of jobs) is still valid and therefore, the sub-problem do not need to be solved again. The second reason is that if the number of jobs in the original problem is large, there is no reason that the number of jobs in each sub-problem be large too. Indeed, the number of jobs in each sub-problem is equal to the number of jobs in the queue of the specified machine at the instant when the sub-problem is generated. Thus, only a few jobs in the original problem are considered in each sub-problem. So the number of jobs in the most of sub-problems is much smaller than the number of jobs in the original problem.

5. Design of Test Problems

In order to evaluate the performance of the proposed method a numerical experiment is performed. For this purpose, the performance of the proposed method is compared with SPT rule in industry-size scheduling problems. The rationale behind this comparison is based on the results of research carried out by Barret and Kadipasaoglu [15] and Rajendran and Holthaus [16]. Barret and Kadipasaoglu [15] perform a simulation study and evaluate the performance of nine dispatching rules in dynamic flow shop for various objective functions. They study five static and four dynamic dispatching rules. They conclude that SPT rule performs the best among the nine studied dispatching rules in terms of mean flow time, mean lateness and percentage of tardy jobs. Rajendran and Holthaus [16] conduct a comparative study on the performance of dispatching rules for dynamic flow shops and job shops. They consider 13 dispatching rules for various objective functions. Some of these studied dispatching rules are heuristic rules which proposed by the authors or previous research. They observe that SPT rule is the overall most effective rule among the 13 studied dispatching rules for minimizing the number of tardy jobs in dynamic flow shops and job shops.
Different kinds of dispatching rules are examined in these two comparative studies. However, the SPT rule has the better performance than the other dispatching rules, even better than the dynamic dispatching rules and heuristic dispatching rules. Thus, although the SPT rule is a simple method, it has a better performance than the more complicated dispatching rules for our research problem. On the other hand, our proposed research problem has been presented for the first time in the stochastic context and there is not any other algorithm available for solving this problem. Therefore, we have to choose the best policy between available dispatching rules to evaluate our proposed method. To the best of our knowledge, the SPT rule is the most effective one for minimizing the number of tardy jobs in dynamic flow shop presented in literature so far. Therefore, to test the performance of the proposed method, it is compared with SPT rule in several test problems.
Two factors are considered to generate test problems for this research. The first important factor is the size of test problems which is determined by the number of jobs and number of machines. Lodree *et al.* [14] performed their experiments by considering test problems including 10-50 jobs and 2-10 machines in a cell. Using this as a guideline, the same number of jobs and number of machines are used in this research. Test problems based on the number of jobs are classified in three different levels: small, medium, and large. The problems include 10, 20, and 50 jobs are classified as small, medium, and large size, respectively. Similarly, test problems based on their number of machines are classified in three different levels: small, medium, and large. The problems include 2, 5, and 10 machines are classified as small, medium, and large size, respectively.
The second important factor to generate test problems is the shop condition. Lodree *et al.* [14] categorize the shop condition to “high shop” and “low shop”. High shop condition represents a very dynamic environment. In this environment jobs arrive during a long interval after beginning the planning horizon, due date of jobs are tight and vary during a long interval. So, the expected number of tardy jobs in high shop condition is probably large. However, low shop condition represents less dynamic environment. In this environment jobs arrive only during a short interval close to the beginning of the planning horizon, the due date of jobs are loose and vary during a short interval. So, the expected number of tardy jobs in low shop condition is probably small. Using Lodree *et al.* [14] as a guideline, two shop conditions are considered for evaluation of the proposed method. We control the parameters of test problems using the method of Hariri and Potts [17] in order to simulate the various shop conditions.

Therefore, three levels of number of jobs, three levels of number of machines, and two levels of shop condition lead to 18 classes of test problems. For each class, one numerical experiment is generated with the following settings.
For each job j and each machine i an integer processing time p_{ij} is generated based on Uniform Distribution $U(1,100)$. Using method of Hariri and Potts [17], we simulate the low shop and high shop conditions. In this method, the maximum completion time P is estimated at first by identifying with the largest processing time requirement as follows:

$$P = \left[\sum_{i=1}^{m}\sum_{j=1}^{n} p_{ij} + (n+1)\max_{i\in[1,n]}\left\{\sum_{j=1}^{n} p_{ij}\right\}\right] \quad (13)$$

Based on method proposed by Hariri and Potts [17], two parameters α and β are used to control the tightness of due dates. The parameters α and β are lower and upper bounds on mean due dates. Then, the means of due dates μ_j are generated randomly and independently based on Uniform Distribution with parameters $(P\alpha, P\beta)$. We use $\alpha = 0.3$ and $\beta = 0.9$ for high shop condition and $\alpha = 0.8$ and $\beta = 1$ for low shop condition. The values of parameters for shop conditions is selected based on our extensive experiments, in which various parameters were tested, and the values which differentiate the low shop condition from high shop condition best are selected. The variance of due dates are generated based on Uniform Distribution such that 99% of due dates to be positive. Therefore, for a given μ_j, σ_j is generated randomly from Uniform Distribution with parameters $(0, \mu_j/2.33)$. In a similar way, the parameters δ and ε are loweand upper bounds on release dates. Therefore, the release dates r_j are generated randomly and independently based on Uniform Distribution with parameters $(P\delta, P\varepsilon)$. We use $\delta = 0$ and $\varepsilon = 0.25$ for high shop condition and $\delta = 0$ and $\varepsilon = 0.05$ for low shop condition.

6. Computational Results

The proposed method and SPT rule are coded in C# programming language on the platform of Microsoft .Net Framework 3.5. The LINGO Release 11.0 [18] is used to solve the sub-problems. Once a sub-problem is generated, the original code calls LINGO and provides the values of the parameters. Then, LINGO solves the sub-problem and provides the optimal solution of the sub-problem. Computational results are obtained by 2GHz Intel® Core™ 2 Duo laptop with 3GB memory and Microsoft Windows XP Professional operating system. In both proposed method and SPT rule, once the due date of a job has passed, that job is considered as a tardy job and its processing on downstream machines is ignored.
The expected number of tardy jobs is computed for the proposed method by using both linear programming (LP) and non-linear programming (NLP) model and compared with the results of the SPT rule. For each numerical experiment, 10,000 replications are run in order to have a good estimation of the expected number of tardy jobs. Each replication is generated randomly based on described method in previous section. The results based on different size of test problems and shop conditions are summarized in Table 1.

Note that for each method $\mathrm{E}\left[\sum U_j\right]$ represents the average number of tardy jobs and the CPU time represents the average computation time (in seconds), over 10,000 replications. In addition, the percentage difference of expected number of tardy jobs obtained by proposed method using linear and non-linear programming model from SPT rule is calculated.

The performance of the proposed method (with two different approaches for solving the sub-problems) and the SPT rule are compared based on an experimental design. The design of experiments should address if any of these methods is capable of identifying a statistically significant smaller expected number of tardy jobs. In order to perform an objective experiment, a randomized block design is selected. Different classes of test problems are source of variability in this experiment. So, blocking is used to eliminate its effect on the statistical comparisons among the three methods. Hence, each class of test problem is considered as blocking factor.

The analysis of variance (ANOVA) conducted using the software SPSS Release 16.0 [19] provides a P-value of 0.004. At a significance level (α) of 0.05, this indicates that there is statistically significant difference among the expected numbers of tardy jobs obtained by the three methods. Further testing based on Tukey's test (Montgomery [20]) at $\alpha = 0.05$ shows that the proposed method using linear programming model performs better than SPT rule, but significant difference cannot be identified between SPT rule and the proposed method using non-linear programming model. The results of ANOVA table and post-hoc analysis with Tukey's test are presented in Table 2 and Table 3, respectively.

Table 1: Comparison of the proposed method with SPT rule

Size of Test Problem ($n \times m$)	Shop Condition	SPT		Proposed Method Using LP Model			Proposed Method Using NLP Model		
		$E[\sum U_j]$	CPU Time	$E[\sum U_j]$	CPU Time	% difference with SPT	$E[\sum U_j]$	CPU Time	% difference with SPT
10×2	Low	1.25	0.0	1.25	0.1	1.2%	1.25	0.1	-1.4%
	High	2.88	0.0	2.69	0.1	7.4%	2.67	0.1	8.1%
10×5	Low	2.07	0.0	1.99	0.1	4.1%	2.12	0.1	-2.3%
	High	5.78	0.0	5.22	0.1	10.8%	5.30	0.1	9.1%
10×10	Low	3.30	0.0	3.18	0.1	3.8%	3.26	0.2	1.4%
	High	8.42	0.0	7.71	0.1	9.1%	7.90	0.1	6.5%
20×2	Low	2.16	0.0	2.11	0.4	2.6%	2.25	0.3	-3.6%
	High	4.60	0.0	3.75	0.3	22.6%	4.02	0.3	14.5%
20×5	Low	3.30	0.0	3.18	0.6	3.7%	3.25	0.6	1.4%
	High	7.81	0.0	5.76	0.4	35.6%	6.29	0.4	24.3%
20×10	Low	5.22	0.0	4.88	0.8	7.0%	4.97	0.9	5.2%
	High	13.19	0.0	10.17	0.5	29.7%	10.95	0.6	20.4%
50×2	Low	4.95	0.0	4.60	4.0	7.5%	5.32	2.4	-6.9%
	High	10.22	0.0	6.81	3.1	50.0%	9.30	1.9	9.9%
50×5	Low	6.73	0.0	6.02	8.6	11.7%	7.30	6.1	-7.9%
	High	14.95	0.0	7.91	5.2	89.0%	11.74	4.8	27.4%
50×10	Low	9.88	0.0	8.16	12.6	21.1%	9.28	10.0	6.5%
	High	21.16	0.0	9.93	7.6	113.1%	15.45	6.7	37.0%
Average		**7.10**	**0.00**	**5.30**	**2.48**	**23.8%**	**6.26**	**1.97**	**8.3%**

Table 1: ANOVA table for comparison of the three methods

Source	Type III Sum of Squares	df	Mean Square	F	Sig.
Corrected Model	798.459[a]	19	42.02	18.941	0
Intercept	2088.8	1	2089	941.47	0
Method	29.434	2	14.72	6.633	0
Problem Size	769.025	17	45.24	20.389	0
Error	75.434	34	2.219		
Total	2962.69	54			
Corrected Total	873.894	53			

Table 2: Tukey's test for simultaneous paired comparisons of methods

Source	Type III Sum of Squares	df	Mean Square	F	Sig.
Corrected Model	798.459[a]	19	42.02	18.941	0
Intercept	2088.8	1	2089	941.47	0
Method	29.434	2	14.72	6.633	0
Problem Size	769.025	17	45.24	20.389	0
Error	75.434	34	2.219		
Total	2962.69	54			
Corrected Total	873.894	53			

Understandably, the proposed method using linear programming model performs better than SPT rule in all class of test problems. The better performance of proposed method using linear programming model compared with the one using non-linear programming model indicates that the linear programming model leads to better solutions for sub-problems. Therefore, estimation of sub-problems solutions by linear programming model leads to closer optimal solutions than local optimum solutions obtained by non-linear programming model. It is important to note that as the size of problems increases, the performance of the proposed method improves. In addition, the proposed method demonstrates a much more improvement than SPT rule in high shop conditions compared with low shop conditions. The minimum difference of proposed method using linear programming model and SPT rule is 1.2% while the maximum difference of these two methods is 113.1%. Consequently, the better performance of proposed method using linear programming model is more conspicuous in hard problems. Briefly, the proposed method using linear programming model performs 23.9% better than SPT rule on average for industry-size scheduling problems. The performance of the proposed method using linear programming model is 7.0% and 40.8% better than SPT rule on average in low shop conditions and high shop conditions of industry-size scheduling problems, respectively. Thus, the proposed method using linear programming model is the preferred method for solving the research problem.

7. Concluding Remarks

This research presents a dynamic method for solving the stochastic flow shop problems with minimizing the expected number of tardy jobs. This research considers the stochastic dynamic flow shop with deterministic processing times and stochastic due dates with normal distribution. A dynamic scheduling method is proposed for this problem that performs averagely 23.9% better than SPT rule which is currently the most effective rule for minimizing the number of tardy jobs in dynamic flow shop for industry-size scheduling problems. The computational study shows that the proposed method has salient performance in hard problems, i.e., the larger the size of problem or the more dynamic the shop condition, the better the performance of the proposed method than SPT rule. For instance, the proposed method can produce results which are 1.2% better than SPT rule in an easy problem to over 113% better than SPT rule in a hard problem. The proposed method needs more complex calculations than SPT rule but, the further calculations are not that significant. Yet, the large-size problems can be solved within reasonable time by the proposed method. However, CPU time of the proposed method and SPT rule does not make any significant difference in small-size problems. Therefore, the complex calculations of the proposed method can be justified because of the much better performance of the proposed method compared with SPT rule.

There are several directions for future research to extend the proposed method. The first one is to develop a method for the case of this problem in which the processing times are stochastic or machines have random breakdowns. Another extension is to generalize the proposed method for the case in which jobs have different weights.

References

1. Pinedo, M.L., 2008, Scheduling: Theory, Algorithms, and Systems, 3rd Edition, Prentice Hall, New York.
2. Moore, J.M., 1968, “An n job, one machine sequencing algorithm for minimizing the number of late jobs”, Management Science, 15, 102-109.
3. Balut, S.J., 1973, “Scheduling to minimize the number of late jobs when setup and processing times are uncertain”, Management Science, 19, 1283-1288.
4. Kise, H. and Ibaraki, T., 1983, “On Balut's algorithm and NP-completeness for a chance constrained scheduling problem”, Management Science, 29, 384-388.
5. Pinedo, M.L., 1983, “Stochastic scheduling with release dates and due dates”, Operations Research, 31, 559-572.
6. Boxma, O.J. and Forst, F.G., 1986, “Minimizing the expected weighted number of tardy jobs in stochastic flow shops”, Operations Research Letters, 5, 119-126.
7. Emmons, H. and Pinedo, M.L., 1990, “Scheduling stochastic jobs with due dates on parallel machines”, European Journal of Operational Research, 47, 49-55.
8. Sarin, S.C., Erel, E. and Steiner, G., 1991, “Sequencing jobs on a single machine with a common due date and stochastic processing times”, European Journal of Operational Research, 51, 188-198.
9. De, P., Ghosh, J.B. and Wells, C.E., 1991, “On the minimization of the weighted number of tardy jobs with random processing times and deadline”, Computers & Operations Research, 18, 457-463.
10. Jang, W., 2002, “Dynamic scheduling of stochastic jobs on a single machine”, European Journal of Operational Research, 138, 518-530.

11. Seo, D.K., Klein, C.M. and Jang, W., 2005, “Single machine stochastic scheduling to minimize the expected number of tardy jobs using mathematical programming models”, Computers & Industrial Engineering, 48, 153-161.
12. van den Akker, J.M. and Hoogeveen, J.A., 2008, “Minimizing the number of late jobs in a stochastic setting using a chance constraint”, Journal of Scheduling, 11, 59-69.
13. Trietsch, D. and Baker, K.R., 2008, “Minimizing the number of tardy jobs with stochastically-ordered processing times”, Journal of Scheduling, 11, 71-73.
14. Lodree, E., Jang, W. and Klein, C.M., 2004, “A new rule for minimizing the number of tardy jobs in dynamic flow shops”, European Journal of Operational Research, 159, 258-263.
15. Barrett, R. and Kadipasaoglu, S., 1990, “Dispatching rules for a dynamic flow shop”, Production and Inventory Management Journal, First Quarter, 54-58.
16. 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.
17. Hariri, A. and Potts, C., 1989, “A branch and bound algorithm to minimize the number of late jobs in a permutation flow shop”, European Journal of Operational Research, 38, 228-237.
18. LINGO Release 11.0, 2008, Lindo Systems Inc., Chicago: USA.
19. SPSS Release 16.0, 2008, SPSS Inc., Chicago: USA.
20. Montgomery, D.C., 1996, Design and Analysis of Experiments, 4th Edition, John Wiley & Sons, New York.