International Journal of Innovative
Computing, Information and Control
Volume **10**, Number **6**, December **2014**

ICIC International ©2014 ISSN 1349-4198
pp. 2113–2126

MULTI-OBJECTIVE DYNAMIC JOB SHOP SCHEDULING: A SURVEY AND PROSPECTS

Xiaoning Shen, Min Zhang and Jingzhi Fu

School of Information and Control
Nanjing University of Information Science and Technology
No. 219, Ningliu Road, Nanjing 210044, P. R. China
sxnystsyt@sina.com

Received January 2014; revised May 2014

Abstract. *Job shop scheduling is a classical combinational optimization problem. Most real-world environments are dynamic and a variety of disruptions may occur unexpectedly, e.g., arrivals of new jobs, machine breakdowns. This paper analyzes the limitations of the static approaches when rescheduling in dynamically changing environments, and the instability problem induced by rescheduling when only based on shop efficiency. Then issues that have appeared in recent years on dynamic job shop scheduling with multiple objectives are described. We categorize the disruptions as data uncertainties and real-time events, and classify the typical objectives in dynamic scheduling into three categories, which are the shop efficiency, schedule robustness and system stability. A new mathematical model of the multi-objective dynamic job shop scheduling problem (MODJSSP) is constructed based on a robust-reactive scheduling approach. A literature review of the state-of-the-art multi-objective dynamic job shop scheduling approaches is provided, and the strength and weakness of different approaches are discussed, and the gaps in current research work are summarized. Experimental studies validate the effectiveness and efficiency of our new mathematical model for MODJSSP.*
Keywords: Job shop scheduling, Dynamic scheduling, Multi-objective optimization, Evolutionary algorithm, Mathematical modelling

1. **Introduction.** Job shop scheduling problem (JSSP) is well-known as a strongly NP-hard combinational optimization problem [1]. In the job shop, suppose there are n jobs to be processed on m machines. Each job J_i contains a sequence of n_i operations $(O_{i1}, \cdots, O_{ij}, \cdots, O_{in_i})$. Each operation O_{ij} should be processed on a predefined machine M_k ($M_k \in \{1, 2, \cdots, m\}$) without interruption within a given processing time p_{ijk}. The operations of each job should be processed one by one in the given order without preemption, and at most one operation can be processed on each machine at one time [2]. The task of JSSP is to find out the best sequences for processing jobs on each machine to achieve the required objectives (e.g., the shortest makespan) subject to certain constraints.

Until recently, scheduling problems which are static in nature were studied assuming that all the problem information is known beforehand. However, environments in a real-world job shop are usually changing dynamically as unpredictable real-time events take place during the process of production, e.g., one machine breaks down suddenly and gets repaired after some time, new urgent jobs arrive in a stochastic way. A previously optimal schedule may become ineffective or infeasible in the new manufacturing environment. Moreover, some information about the job shop is not given beforehand. For example, the due date, release date, and processing time of each new job are often not given until the job arrives. Such problems are generally known as dynamic scheduling [3,4].

Multiple objectives have to be considered simultaneously in JSSP. Most of the existing research generated a new schedule by minimizing the shop efficiency like makespan, tardiness, or flow-time [5,6]. However, it may produce a new schedule totally different from the previous one. Some remaining operations in the original schedule which have not begun processing when rescheduling may have their starting time brought forward or delayed, which has a negative impact on other production activities planned based on the original schedule, and brings instability and lack of continuity in the shop system [4]. Besides, since it is often the case that some parameters in the job shop such as the operation processing time are uncertain and change over time, the robustness of the schedule efficiency performances to such data variances should also be taken into account together with the efficiency and stability. Above all, the job shop scheduling in the real world has both the dynamic and multi-objective nature.

Most existing literature on the multi-objective dynamic job shop scheduling problem (MODJSSP) adopted the linear weighted sum approach to handle multiple objectives [3-9]. However, in most real-world cases, it would be difficult to identify suitable weights for each objective. Furthermore, multiple objectives are usually conflicting with each other. It is better to handle multiple objectives with a full knowledge about their Pareto front, and obtain various trade-offs among different objectives.

The papers on MODJSSP were scattered among different journals and conferences. No systematic survey is available in the field of dynamic JSSP with multiple objectives. The primary purpose of this paper is to define MODJSSP mathematically, review the state-of-the-art in MODJSSP, discuss the strength and weakness of different existing approaches, and identify future research directions. The rest of this paper is organized as follows. Section 2 provides a categorisation of the dynamic features and optimization objectives in JSSP. Section 3 gives a definition of MODJSSP mathematically. In Section 4, a literature review of the state-of-the-art scheduling approaches for MODJSSP is presented. Section 5 points out gaps of the existing work and the future research directions. Section 6 gives experimental studies. Finally, conclusions are drawn in Section 7.

2. Dynamic Features and Multiple Objectives in the Job Shop Scheduling.

2.1. Dynamic features in the job shop scheduling.

Environments of the real-word manufacturing system are dynamic, and a typical job shop is always affected by some disruptions. In this paper, disruptions are divided into two main categories: data uncertainties and real-time events.

(i) Data uncertainties. A part of the system's information may be uncertain and changes over time. A typical example is that the processing time of an operation depends on the current machine condition and job specification when the operation is being performed, so it cannot be determined beforehand. Another example is that the due date of a job may change dynamically since a less important job today may be of high importance tomorrow. Changes in the job order priority or batch size also belong to information uncertainties.

(ii) Real-time events. This kind of events may occur in two ways. In the first way, real-time events occur haphazardly during the execution of the job shop, so the non-operation-disruption assumption in the static JSSP is violated. For instance, one machine breaks down suddenly. In the second way, dynamic events occur unpredictably, such as jobs arrive in a continuous and stochastic pattern.

Scheduling in job shop environments with uncertainties and occurrences of real-time events is called the dynamic job shop scheduling problem (DJSSP).

2.2. **Multiple objectives in the dynamic job shop scheduling.** In a lot of real-world manufacturing systems, many criteria should be optimized simultaneously, which are often conflicting with each other. It is necessary to learn how the different trade-offs among multiple objectives can be obtained.

We classify the typical objectives of DJSSP into three categories, which are the shop efficiency, schedule robustness and system stability.

(1) The shop efficiency. There are several kinds of objectives related to the job shop efficiency [7,10]. Among them, completion time based and tardiness based objectives are the most commonly used. Makespan which represents the completion time of all the available jobs in the shop is a typical one that belongs to the completion time based objective. Tardiness gives penalties to delays from the due date of each job. It is usually defined as the weighted sum (or mean) of differences between the completion time and due date of each job in which its completion time is larger than the due date. Other kinds of objectives with regard to the shop efficiency include: minimizing the mean, total, or the maximal flow time, where flow time is the time interval between the arrival and the departure time of a job; and the machine related objectives which consist of maximizing the utilization rate of machines, the machine idle time, minimizing the total waiting time.

(2) Robustness. Considering data uncertainties in the DPSP, robustness which measures the sensitivity of the schedule quality to disturbances should also be taken into account. The definition of the robustness measure can be divided into two categories: scenario based approaches and surrogate measures. In scenario based approaches [11-13], a set of scenarios are constructed by sampling the values of uncertain parameters, and then the robustness measure is defined as the deviation between the performances obtained from different scenarios and that produced in the initial deterministic scenario. Scenario based approaches are able to find a solution with strong robustness, especially when the sample size is large. However, as indicated by [14], exhausted simulations may be time-consuming. To deal with this problem, some surrogate measures are designed to approximate the robustness. The average float time of all job operations and the total sum of free slacks were adopted in [15,16] to evaluate the robustness of a schedule, respectively. In [17], a modification of the slack based robustness measure was proposed to utilize some uncertainty information that was available beforehand, such as probability of the machine breakdown occurrence, etc. With the idea that robust solutions were expected to be located on broad peaks, [18] defined a neighborhood based robustness measure where a weighted average of the makespans of schedules located in the predefined neighborhood of a schedule was calculated.

(3) Stability. In the rescheduling mode, a new schedule is regenerated once a real-time event occurs or rescheduled periodically at fixed time intervals. Stability measures the deviation between the revised and original schedules, which is used to quantify the negative impact caused by modifying the previous schedule [15,19]. There is no universal definition for stability [4]. [20] defined stability as the number of times the rescheduling occurred and suggested that more frequent rescheduling indicates a less stable schedule. In [19,21], stability was defined as the operation starting time deviation and operation sequence deviation. In [3,4], stability had two dimensions. One was the starting time deviation, and the other reflected how close to the current time changes were made, which had to use an approximate function to associate a penalty with rescheduling jobs to earlier time.

3. **Definitions of the Multi-objective Dynamic Job Shop Scheduling Problem.** MODJSSP deals with finding out the best sequences for processing jobs on each machine in an uncertain and dynamic environment, with the aim to optimize multiple objectives simultaneously subject to certain constraints. So far, only a few studies have focused on the mathematical formulations for the dynamic JSSP. A mathematical model for dynamic scheduling in a flexible job shop which minimized a weighted sum of two objectives (makespan and stability) was developed in [3]. It used binary variables to form constraints. [22] presented a neural network model for solving the dynamic hybrid flow shop problem. [23] proposed a robust mathematical model for the job shop scheduling problem with processing time uncertainties. To deal with both data uncertainties and real-time events, we model MODJSSP based on a new robust-reactive approach. A predictive schedule is generated at the initial time t_0 of the job shop by robust scheduling to reduce the schedule quality sensitivity to information uncertainties, and then during the implementation of the schedule, once a dynamic event occurs, a rescheduling approach is triggered to regenerate a new schedule. The time at which a new schedule is constructed is called the rescheduling point, and the time span between two successive rescheduling points is named the rescheduling interval.

Based on the robust-reactive approach, the definition for MODJSSP at a specific rescheduling point t_l ($t_l > t_0$) is given as follows.

At t_l, all the current information gathered from the job shop floor is considered:

- $m(t_l)$ available machines which can work normally at t_l;
- $n(t_l)$ available jobs at t_l, and each job $J_i(t_l)$ $(i = 1, 2, \cdots, n(t_l))$ contains $n'_i(t_l)$ available operations $O_{ij}(t_l)$ (suppose $G_i(t_l)$ is the index of the first unprocessed operation in job $J_i(t_l)$ at t_l, then $j = G_i(t_l), G_i(t_l) + 1, \cdots, G_i(t_l) + n'_i(t_l) - 1$). The operation $O_{ij}(t_l)$ is regarded as available at t_l if the following three conditions are satisfied simultaneously: (i) $O_{ij}(t_l)$ has not begun processing by t_l; (ii) the machine on which $O_{ij}(t_l)$ is predefined to be performed can work normally at t_l; and (iii) all the unfinished operations preceding $O_{ij}(t_l)$ in the job $J_i(t_l)$ satisfy the above condition (ii). The job which has available operations left is called an available job.

A new schedule which represents the sequences for processing job operations on each machine is constructed by optimizing the following objectives:

$$\text{optimize } \mathbf{F} = [f_1(t_l), f_2(t_l), f_3(t_l)] \tag{1}$$

where $f_1(t_l)$ represents the objective related to the shop efficiency, e.g., the makespan which means the elapsed time required for finishing all the available jobs rescheduled at t_l; $f_2(t_l)$ is the schedule robustness measure; and $f_3(t_l)$ denotes the system stability. Note that any criteria required by the real-world job shop can be added as one of the objectives.

In MODJSSP, constraints to the search space change dynamically with the occurrences of random events and uncertainties. They are listed as follows.

1) Technological constraints

An operation of a job can be performed by only one type of machine which is predefined. For instance, the j^{th} operation of the i^{th} job ($O_{ij}(t_l)$) is predetermined to be processed on the k^{th} available machine $M_k(t_l)$ $(k = 1, 2, \cdots, m(t_l))$ at t_l.

2) Processing time constraints

Assume the operation $O_{ij}(t_l)$ is predefined to be processed on the machine $M_k(t_l)$, then a processing time p_{ijk} is associated with $O_{ij}(t_l)$. If the operation processing time is uncertain and changes overtime, then a value p_{ijk}^{est} is estimated as the initial scenario.

3) Initial available time constraints

For each available job $J_i(t_l)$ at t_l, assume $c_{i(G_i(t_l)-1)}$ denotes the completion time of the last operation of job $J_i(t_l)$ that has begun processing before t_l, then the initial release time of job $J_i(t_l)$ during the rescheduling interval of t_l is:

$$R_i(t_l) = \max(t_l, c_{i(G_i(t_l)-1)}), \text{ for } i = 1, 2, \cdots, n(t_l) \tag{2}$$

For each available machine $M_k(t_l)$ at t_l, assume $c^{k\_\mathrm{last}}(t_{l-1})$ denotes the completion time of the last operation processed on $M_k(t_l)$ before t_l, then the initial available time of machine $M_k(t_l)$ during the rescheduling interval of t_l is:

$$A_k(t_l) = \max\left(t_l, c^{k\_\mathrm{last}}(t_{l-1})\right), \text{ for } k = 1, 2, \cdots, m(t_l) \tag{3}$$

Equation (2) gives the initial release time of each job. It guarantees that all the operations that have begun processing before t_l not be considered in the rescheduling model. If $G_i(t_l) = 1$, then $c_{i(G_i(t_l)-1)} = 0$. Equation (3) gives the initial idle time of each machine. It indicates that one machine is available until it has finished all the operations that have begun before t_l.

4) No preemption constraints

An operation of a job cannot be processed until its preceding operations are completed.

Assume $O_{iG_i(t_l)}(t_l)$ represents the first unprocessed operation of job $J_i(t_l)$. It should start after the job initial release time $R_i(t_l)$, and its starting time $s_{iG_i(t_l)}(t_l)$ satisfies:

$$s_{iG_i(t_l)}(t_l) \geq R_i(t_l), \text{ for } i = 1, 2, \cdots, n(t_l) \tag{4}$$

For other unprocessed operations of job $J_i(t_l)$ at t_l, the starting time $s_{ij}(t_l)$ satisfies:

$$s_{ij}(t_l) \geq c_{i(j-1)}(t_l), \text{ for } i = 1, 2, \cdots, n(t_l), \quad j = G_i(t_l) + 1, \cdots, G_i(t_l) + n'_i(t_l) - 1 \tag{5}$$

where $c_{ij}(t_l)$ represents the completion time of operation $O_{ij}(t_l)$.

An operation can be processed on its predefined machine until the machine has finished its previously scheduled operations. Suppose at t_l, $O_{ij}(t_l)$ is scheduled as the r^{th} operation on $M_k(t_l)$.

If $r = 1$, then $O_{ij}(t_l)$ should start after the initial machine available time $A_k(t_l)$:

$$s_{ij}(t_l) \geq A_k(t_l), \text{ for } r = 1, \quad k \in \{1, 2, \cdots, m(t_l)\} \tag{6}$$

If $r \geq 2$, suppose the completion time of the $(r-1)^{\text{th}}$ operation scheduled on $M_k(t_l)$ is $c^{O^{k(r-1)}}(t_l)$, then

$$s_{ij}(t_l) \geq c^{O^{k(r-1)}}(t_l), \text{ for } r = 2, \cdots, q^k(t_l), \quad k \in \{1, 2, \cdots, m(t_l)\} \tag{7}$$

where $q^k(t_l)$ denotes the number of operations scheduled on the machine $M_k(t_l)$ at t_l.

It should be mentioned that at the initial time t_0, the mathematical model is similar to the above model for the rescheduling point $t_l > t_0$. The difference is that only objectives $f_1(t_l)$ and $f_2(t_l)$ related to the shop efficiency and robustness are to be optimized.

4. **Multi-objective Dynamic Job Shop Scheduling Approaches.**

Two primary issues in MODJSSP are how to deal with uncertainties and dynamic events when they occur during the implementation of an original schedule, and how to optimize multiple objectives simultaneously. Generally, representative approaches in the existing literature on MODJSSP can be classified into four categories described as follows.

4.1. **Alternative approaches for heuristic rules.** In completely reactive scheduling [24-26], no deterministic schedule is produced a priori, and only partial schedules are created for the immediate future based on current job shop status and local information at each decision instant. The simple priority rules are commonly used for the local decision making, such as the shortest processing time (SPT) [27], first in first out (FIFO), and earliest due date (EDD) [28]. For example, when a machine becomes idle, the job with the highest priority will be selected from the candidates in the waiting queue based on a priority dispatching rule, or when a new job arrives, its due date is estimated by a due-date assignment rule. The simple priority rules are usually intuitive and easy to implement. However, most of them are developed to meet just one production requirement, thus they are unable to deal with multiple objectives simultaneously. Some work has been done to design an alternative approach for the local decision making instead of the simple priority rules. A hybrid multiple attribute decision making (MADM) technique which used grey numbers to deal with uncertainties was given in [8] to determine which lot in the waiting queue was suitable to be processed next when a machine was free. Challenges for this method are how to determine the positive and negative ideal alternatives, and how to combine all criteria into a global value appropriately. [9] focused on the implementation concept of a discrete event simulation based dynamic scheduling system using conjunctive simulated scheduling. The weighted sum method was used to convert the due date priority, set up cost factor, and cycle time priority into a single one, and it was used as the performance measure to select a lot from the waiting queue of each machine at every decision instant. However, just a simple manual simulation experiment with ten jobs and one machine was given. In [7], the artificial immune system was applied to establish the idiotypic network model in advance that encapsulated the relationship between the job shop configurations, objectives and existing dispatching rules based on a large amount of training data. Then at each scheduling point, an existing dispatching rule that performed the best under the given objective and a specific environment condition was determined by looking up the idiotypic network model, and calculating the concentration to schedule the waiting jobs.

There are three common limits with the existing approaches in this category. Firstly, multiple objectives are handled by the linear weighted sum method, where it would be difficult to identify suitable weights for each objective in most real-world cases. Secondly, most studies focus on the job dispatching method. More approaches for surrogating other heuristic rules such as the due-date assignment rule should be further studied. Thirdly, this class belongs to completely reactive scheduling, which makes decisions locally based on attributes of current jobs and machines, without considering global information. Thus the whole job shop performance cannot be guaranteed, and it is easy to be trapped into a local optimum. It cannot offer any plan information to other production activities [24], e.g., when to deliver the secondary materials such as tools to the corresponding machines.

This class of approaches is suitable for online dynamic scheduling.

4.2. **Evolution of the customized heuristic rules.** In completely reactive scheduling, although a lot of human-made simple priority rules have been widely applied in industry, little is known about the robustness of such rules to the changing shop environment [5], and their efficiency performance remains poor [29]. So it is difficult for a scheduler to decide which rule should be used in a specific job shop configuration and operation state. In recent years, more attention has been paid to evolving customized and composite heuristic rules based on the characteristics of a specific dynamic environment in advance. Due to its capability to represent and search the program structure, genetic programming (GP) has become the most commonly used tool for evolving heuristic rules [30-34]. Among the existing work, multiple objectives were addressed in several studies when producing the rules. Four multi-objective genetic programming based hyper-heuristic approaches were developed in [35] for simultaneous design of dispatching rules for sequencing and due-date assignment rules for new arriving jobs. Objective evaluations were performed by applying the rules into four different job shop training scenarios with real-time job arrivals and calculating the average performance measures. Reusability of the obtained Pareto rules was validated in another set of testing scenarios with the same type of dynamic events. Gene express programming [36] was adopted in [37] to evolve machine assignment rules and job dispatching rules together in the dynamic flexible job shop scheduling problem where jobs arrive over time. Although three measure criteria of makespan, mean flow time and mean tardiness were considered, they were not dealt with simultaneously, and only one of them was taken as the single objective in each experiment. In [29], a composite dispatching rule which can minimize the weighted sum of makespan, mean tardiness and flow time in the flexible job shop is evolved by GP. The key issue in GP is the selection and design of the terminal set and function set which can meet the requirement of the considered problem.

Problems with this kind of approaches are that individual evaluations are highly computational expensive since a great many training sets with different dynamic job shop scenarios should be used to evaluate the quality of a solution. Besides, offline analysis can become inaccurate over the longer term when facing the real-world shop circumstances with different kinds of unforeseen disruptions from the training job shop instances.

This class belongs to the offline dynamic scheduling approach.

4.3. **Rescheduling in dynamic environments.** This class of approaches can be categorized as the predictive-reactive scheduling [24], where a production schedule is generated first considering the given deterministic information, and then the original schedule is revised in response to a dynamic event to minimize its impact on the system performances in an event-driven way, or the previous schedule is periodically rescheduled at predefined time intervals. Predictive-reactive scheduling uses global information of the shop floor. Thus its search space is large, and high quality schedules which achieve good system performances can be obtained. It is the most commonly used scheduling method in manufacturing systems [25].

A conventional genetic algorithm (GA) was used in [6] to regenerate a new schedule whenever a dynamic event took place, considering the mean job cost and job tardiness. However, it is often inefficient to restart the optimization process with a totally new population [38] due to the heavy computational time cost by a GA. Besides, although the symbolic representation of the schedule individual in [6] is intuitive, it is not easy to handle when programming. In [5], the variable neighborhood search (VNS) was triggered to optimize the makespan and tardiness in respond to a random event, and the parameters in VNS were adjusted by a trained artificial neural network (ANN) according to current job shop conditions at any rescheduling point. The difficulty with this method is how well the weights of ANN are trained is uncertain, because there can be training errors due to the presence of improper or inadequate learning samples. The proposed approaches in [5,6] were compared to some simple priority dispatching rules in various simulated job shops, respectively, and it was found that both of them had better scheduling performances. However, they may disrupt system stability and continuity if a new and totally different schedule is regenerated. In order to solve the instability problem induced by unrestricted rescheduling, bi-objectives of stability and efficiency were considered simultaneously in [3,4]. Both studies used a GA to generate schedules at each rescheduling point. [4]

employed periodic rescheduling, and discussed the influence of the stability term and the length of the scheduling interval on the performance of the methodology used.

Three common weaknesses can be found in the existing work of this category. Firstly, since a variety of data uncertainties exist in the practical job shop, robustness should also be considered in addition to the efficiency and stability objectives when rescheduling so that the sensitivity of the schedule quality to date variances can be reduced. Secondly, with regard to the multi-objective handling method, the linear weighted sum method was usually adopted [3-6] to covert multiple objectives into a single objective. To better deal with multiple objectives, other multi-objective optimization approaches to the field of rescheduling should be further studied. Thirdly, in a large-scale problem, if real-time events occur frequently, how to find the scheduling solutions efficiently within a reasonable time span by an iteration-based algorithm like GA is an open question.

This class of approaches is suitable for online dynamic scheduling.

4.4. **Generation of a priori robust schedule.** The class can be categorized as the robust scheduling, which develops a predictive schedule in advance that can minimize the impact of disruptions on the shop performances without rescheduling [39]. [40] introduced four different probability distributions to model stochastic processing times in a permutation flow-shop scheduling problem, and proposed three uncertainty handling approaches which estimated the fitness of a solution based on the single evaluation, average value of several evaluations and probabilistic estimate, respectively. Makespan and tardiness were simultaneously optimized by an evolutionary algorithm. A simplified multi-objective genetic algorithm was proposed in [41] for the multi-objective stochastic job shop scheduling problem with exponential processing time. Objective evaluations and the decoding procedure were executed by operations of exponential random variables. [12,14,18] generated the robust schedule for one machine scheduling, JSSP, and flexible JSSP with random machine breakdowns by a GA, respectively. They tried to obtain a predictive schedule which can best accommodate disruptions and minimize the impact of machine breakdowns on performances of makespan and stability. In [17], a multi-objective evolutionary algorithm is used to solve the flexible job shop problems with random machine breakdowns. Two slack-based surrogate measures for robustness which considered the probability of machine breakdowns and the machine breakdown locations were developed, and compared with the scenario-based and other existing surrogate robustness measures. However, in [12,17], it was assumed that all possible breakdowns were aggregated as one type of breakdown, so the predictive schedule quality may degrade when facing different types of breakdowns. [42] employed a multi-objective immune algorithm to solve JSSP with uniform processing time uncertainties. The mean and standard deviation of makespan obtained from a set of workflow simulations were used as two objectives.

In this category, in order to guarantee the applicability of the obtained schedule to various dynamic job shop conditions, an efficient fitness estimation strategy for dealing with disruptions needs to be designed. Besides, only one type of disruption (processing time or machine breakdown was commonly used) was considered in the stochastic model. Thus, it might become inaccurate when facing unexpected uncertainties or events which were not incorporated in its predictive stochastic model.

This class also belongs to the offline dynamic scheduling approach.

5. **Discussions.** There are some gaps in the existing literature on MODJSSP. Firstly, both data uncertainties and real-time events occurring in the job shop should be addressed, while most of the current work just takes one of them into account. Secondly, shop efficiency, schedule robustness and system stability should be considered simultaneously

when regenerating a new schedule, or selecting an appropriate rescheduling strategy in response to real-time events. More general and efficient robustness and stability measures should be defined. Thirdly, types of uncertainties and dynamic events considered in the existing studies are rather limited (processing time uncertainties, machine breakdowns and new job arrivals are most commonly used), thus more kinds of disruptions (e.g., changes in the job specifications or due dates) should be incorporated in the future experimental studies. Fourthly, some assumptions have been made in the current work to simplify the complexity of JSSP. To bridge the gap between the theoretical and real-world model, more practical constraints in the job shop must be considered, e.g., the sequence dependent setup time, lot sizing, limited buffer, due date and resource availability.

Evolutionary algorithms (EAs) are a class of stochastic optimization approaches that simulate the process of natural evolution. EAs have been recognized to be well suited for multi-objective optimization problems due to their capability to evolve a set of solutions simultaneously in one run. In the past 20 years, multi-objective evolutionary algorithms (MOEAs) received much attention. However, until now, only a few papers have applied MOEAs to MODJSSP [35,40,41].

In the future, it will be interesting to incorporate the problem specific heuristic strategies and preference information into MOEA so that its searching efficiency and convergence speed can be improved when solving MODJSSP. Another promising research topic is to link robust scheduling to the rescheduling approach. In such a way, both the robustness to data uncertainties and the response to dynamic events can be addressed, and the effectiveness of both approaches can be largely improved. The predictive scheduling policies can also serve as a guide for the search of the rescheduling approach.

6. Experimental Studies.

To validate the effectiveness and efficiency of our mathematical model for MODJSSP constructed in Section 3, a realistic job shop has been simulated, and it was compared to the model of [3]. All the experiments in this paper are performed on a personal computer with Intel core i5, 3.2 GHz CPU and 4 GB RAM.

It was pointed out that a job shop with more than six machines presents the complexity involved in large dynamic job shop scheduling [5]. In this paper, a job shop consisting of ten machines is simulated to evaluate the performance of approaches. Simulation starts with a 10×10 static job shop problem, where the initial numbers of jobs and machines are both 10. Then new jobs arrive following the Poisson distribution [31]. Operation processing time uncertainties are also considered because modifications in job specifications may cause the changes in the initially estimated processing time of each operation. Here, processing time variances are assumed to follow the Normal distribution.

In our robust-reactive approach, considering the processing time uncertainties, a predictive schedule is generated at the initial time of the job shop by the MOEA-based robust scheduling, where two objectives of *makespan* and *robustness* are optimized simultaneously. Then during the implementation of the job shop, once a dynamic event occurs (here, a new job arrives), an MOEA-based rescheduling approach is triggered to regenerate a new schedule, where three objectives of *makespan*, *robustness* and *stability* are optimized simultaneously. In contrast, in the model of [3], robustness to uncertainties is not considered, and only *makespan* was optimized in the initial static problem. When rescheduling, *makespan* and *stability* are converted into a single objective and then optimized by a GA. In this paper, the definitions of *makespan* and *stability* are the same as those of [3], while *robustness* is defined as follows:

$$robustness(t_l) = \sqrt{\frac{1}{N}\sum_{q=1}^{N}\left(\max\left(0, \frac{makespan_q(t_l) - makespan_I(t_l)}{makespan_I(t_l)}\right)\right)^2} \tag{8}$$

where $robustness(t_l)$ evaluates the sensitivity of makespan to processing time uncertainties. Here, the scenario-based approach is used. A schedule undergoes a set of processing time scenarios $\{\theta_q | \ q = 1, 2, \cdots, N\}$, where θ_q is the q^{th} sampled scenario of processing times, N is the sample size, and we set $N = 30$. $makespan_I(t_l)$ represents the elapsed time for completing all the jobs rescheduled at t_l under the initially estimated scenario, and $makespan_q$ is the corresponding makespan values under the scenario θ_q.

First, the strength of using the MOEA-based scheduling approach is validated. At each rescheduling point, a set of non-dominated solutions was evolved by an MOEA. In order to demonstrate the trade-offs among these solutions, Pareto fronts evolved at the initial time and at the first rescheduling point (when the first new job arrives) in one run of the MOEA are shown in Figure 1 and Figure 2, respectively. It is obvious from Figure 1 that in the initial static problem, *makespan* is seriously conflicted with the *robustness* objective. When tracing along the Pareto front to find solutions that have higher efficiency (lower *makespan*), it can be observed that *robustness* becomes worse. Table 1 gives several examples of the objective vectors selected from the Pareto front given in Figure 2. A solution may perform well for one objective, but give bad results for others, such as Solution$_1$-Solution$_3$. And one solution may have a 'not bad' value on each objective, which means a good compromise among all the objectives, such as Solution$_4$-Solution$_6$. The Pareto front produced by MOEA can provide the decision maker (e.g., a production scheduler) with a full knowledge about various trade-offs among multiple objectives. It is very helpful for him/her to make an informed decision about the 'best compromise' with regards to his/her preference. It can also help the decision maker adjust his/her preference after he/she understands more about the trade-offs. In contrast, the model of [3] just obtains one solution in one run of GA, which cannot reflect different compromises among multiple objectives. Furthermore, in most real-world cases, it would be difficult to

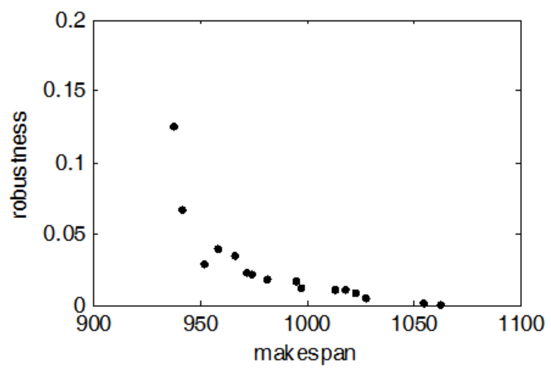

FIGURE 1. Pareto front at the initial time

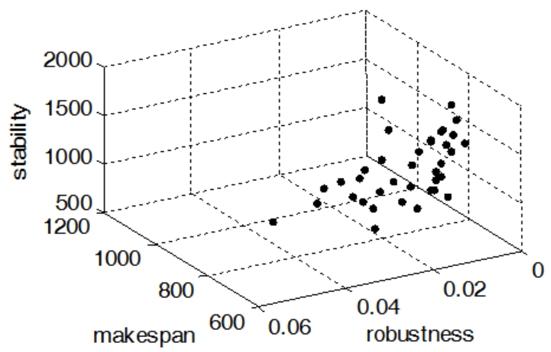

FIGURE 2. Pareto front at the first rescheduling point

TABLE 1. Several examples selected from the Pareto front at the first rescheduling point

	[$makespan_I$, $robustness$, $stability$]
Solution$_1$	[722.1756, 0.0381, 1330.9031]
Solution$_2$	[1138, 0.0001495, 1199.8994]
Solution$_3$	[760.1756, 0.0241, 704.1341]
Solution$_4$	[785.1756, 0.0091, 921.6343]
Solution$_5$	[741.1756, 0.0157, 871.9077]
Solution$_6$	[809.1756, 0.0047, 781.7347]

TABLE 2. Performance improvements (or deteriorations) of MOEA-R over GA-FF on each objective at the initial time (The positive value means improvement and is in bold)

Objective	*makespan*	*robustness*
MOEA-R vs. GA-FF	–6.55%	**64.63%**

identify suitable weights for each objective by using the weighted sum method. Thus, it is better to handle multiple objectives with knowledge about their Pareto front, which can be obtained by our mathematical model that is based on an MOEA-based robust-reactive scheduling approach.

Next, the effectiveness of the initial robust scheduling in improving the schedule robustness to processing time uncertainties is validated. At the initial time of the job shop, performance comparisons were done between our robust approach (two objectives of *makespan* and *robustness* were considered simultaneously, and it is called MOEA-R), and the approach of [3] where only *makespan* was considered (it is called GA-FF). 30 independent runs of both approaches were replicated. Then the value of the objective "*robustness*" was calculated for all the solutions obtained by the two approaches using the same sampled processing times (100 scenarios were sampled at random here), in spite of the fact that only one of them was optimizing this objective. With the aim to determine the overall performance improvement (or deterioration) on each objective by incorporating "*robustness*" as one of the multiple objectives, the solutions of MOEA-R were averaged along each objective, respectively, and also for GA-FF. The quantitative improvement (or deterioration) of MOEA-R over GA-FF on each objective is calculated as follows:

$$Imp_i(t_0) = -\frac{(Avg\_f_i^{\text{MOEA-R}} - Avg\_f_i^{\text{GA-FF}})}{Avg\_f_i^{\text{GA-FF}}} \times 100\%, \quad i = 1, 2 \tag{9}$$

where $Avg\_f_i^{\text{MOEA-R}}$ and $Avg\_f_i^{\text{GA-FF}}$ represent average values of the solutions obtained by MOEA-R and GA-FF on the objective f_i over 30 runs, respectively. The results are listed in Table 2. It can be observed that compared to GA-FF (without *robustness*), our MOEA-R improves the *robustness* significantly with a small sacrifice in the initial *makespan*. The improvement in *robustness* is much more than the deterioration in *makespan*, which indicates that if the predictive schedule is produced by simultaneously considering efficiency (e.g., *makespan*) and robustness, there will be a high opportunity of getting a more robust schedule without seriously affecting the shop efficiency.

7. Conclusions.

This paper gives a comprehensive review of the approaches for solving dynamic job shop scheduling with multiple objectives. Our first contribution is to classify disruptions occurring in the job shop into two categories which are data uncertainties and dynamic events, and identify three classes of typical objectives in dynamic job shop scheduling: shop efficiency, schedule robustness and system stability.

Our second contribution is that we provide a new mathematical definition of MODJSSP, which is based on a robust-reactive scheduling approach. This model can deal with both uncertainties and real-time events, and optimize efficiency, robustness and stability simultaneously. Experimental results show that our method can obtain a Pareto front at each rescheduling point. It provides much better knowledge about various trade-offs in the objective space for the decision maker to make an informed decision, which cannot be achieved by the approaches using a weighted sum method. Furthermore, compared to the approach without considering robustness, our robust scheduling reduces the schedule sensitivity to uncertainties significantly with only a small sacrifice in the makespan under the initial scenario.

Our third contribution is that we review the representative approaches in the existing studies on MODJSSP and classify them into four categories, of which the strength and weakness are discussed, respectively. The first class develops an alternative approach instead of simple priority rules in completely reactive scheduling. It considers only local information thus it may be trapped into a poor local optimum. The second class evolves customized rules based on the features of the environment. However, individual evaluations are highly computational expensive since a lot of training scenarios should be used. In the third class, a new schedule is generated in an event-driven way or periodically. This class may bring instability in job shops. Thus, generating a schedule considering the efficiency, robustness and stability simultaneously is more desirable. The fourth class produces a predictive schedule in advance. Its limitation is that it may become inaccurate over the longer term when facing unforeseen dynamic events.

Our review shows that although some progresses have been made in the field of MODJSSP, some gaps still exist in the current work, and further studies need to be done. To improve the flexibility and robustness of the dynamic scheduling system, the approach which is developed based on the problem specific knowledge, and integrates the merits of different approaches such as heuristics, meta-heuristics, expert systems, and multi-agent architectures would be very promising for the future work. Furthermore, the proposed mathematical model of MODJSSP should be applied to other job shop scenarios with more dynamic events and uncertainties, e.g., machine breakdowns, due-date variations.

Acknowledgement. This work was partially supported by the "Outstanding Young and Middle-Aged University Teachers and Presidents Training Abroad Project" in Jiangsu Province, China, and a Scientific Research Fund of Nanjing University of Information Science and Technology, China (No. S8111127001), through a scholarship to the first author. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] M. R. Garey, D. S. Johnson and R. Sethi, The complexity of flow shop and job shop scheduling, *Mathematics of Operations Research*, vol.1, no.2, pp.117-129, 1976.
[2] J. Gao, L. Sun and M. Gen, A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems, *Computers & Operations Research*, vol.35, no.9, pp.2892-2907, 2008.
[3] P. Fattahi and A. Fallahi, Dynamic scheduling in flexible job shop systems by considering simultaneously efficiency and stability, *CIRP Journal of Manufacturing Science and Technology*, vol.2, no.2, pp.114-123, 2010.
[4] R. Rangsaritratsamee, W. G. Ferrell and M. B. Kurz, Dynamic rescheduling that simultaneously considers efficiency and stability, *Computers & Industrial Engineering*, vol.46, no.1, pp.1-15, 2004.
[5] M. A. Adibi, M. Zandieh and M. Amiri, Multi-objective scheduling of dynamic job shop using variable neighborhood search, *Expert Systems with Applications*, vol.37, no.1, pp.282-287, 2010.
[6] G. Chryssolouris and V. Subramanian, Dynamic scheduling of manufacturing job shops using genetic algorithm, *Journal of Intelligent manufacturing*, vol.12, no.3, pp.281-293, 2001.
[7] X. Qiu and H. Y. K. Lau, An AIS-based hybrid algorithm with PDRs for multi-objective dynamic online job shop scheduling problem, *Applied Soft Computing*, vol.13, no.3, pp.1340-1351, 2013.
[8] S. Yao, Z. Jiang, N. Li, H. Zhang and N. Geng, A multi-objective dynamic scheduling approach using multiple attribute decision making in semiconductor manufacturing, *International Journal of Production Economics*, vol.130, no.1, pp.125-133, 2011.
[9] A. I. Sivakumar and A. K. Gupta, Online multiobjective Pareto optimal dynamic scheduling of semiconductor back-end using conjunctive simulated scheduling, *IEEE Transactions on Electronics Packaging Manufacturing*, vol.29, no.2, pp.99-109, 2006.

[10] Y. Sun, C. Zhang, L. Gao and X. Wang, Multi-objective optimization algorithms for flow shop scheduling problem: A review and prospects, *International Journal of Advance Manufacturing Technology*, vol.55, no.5-8, pp.723-739, 2011.

[11] T. Chaariabc, S. Chaabaneab, T. Loukild and D. Trentesaux, A genetic algorithm for robust hybrid flow shop scheduling, *International Journal of Computer Integrated Manufacturing*, vol.24, no.9, pp.821-833, 2011.

[12] N. Al-Hinai and T. Y. ElMekkawy, Solving the flexible job shop scheduling problem with uniform processing time uncertainty, *World Academy of Science, Engineering and Technology*, vol.64, pp.996-1001, 2012.

[13] H. A. Abbass, A. Bender, H. H. Dam, S. Baker, J. Whitacre and R. Sarker, Computational scenario-based capability planning, *Proc. of the 10th Annual Conference on Genetic and Evolutionary Computation*, Atlanta, GA, USA, pp.1437-1444, 2008.

[14] L. Liu, H. Gu and Y. Xi, Robust and stable scheduling of a single machine with random machine breakdowns, *International Journal of Advanced Manufacturing Technology*, vol.31, pp.645-654, 2007.

[15] V. J. Leon, S. D. Wu and R. H. Storer, Robustness measures and robust scheduling for job shops, *IIE Transactions*, vol.26, no.5, pp.32-41, 1994.

[16] M. A. Al-Fawzan and M. Haouari, A bi-objective model for robust resource-constrained project scheduling, *International Journal of Production Economics*, vol.96, no.2, pp.175-187, 2005.

[17] J. Xiong, L. Xing and Y. Chen, Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns, *International Journal of Production Economics*, vol.141, no.1, pp.112-126, 2013.

[18] M. T. Jensen, Generating robust and flexible job shop schedules using genetic algorithms, *IEEE Transactions on Evolutionary Computation*, vol.7, no.3, pp.275-288, 2003.

[19] S. D. Wu, R. H. Storer and P. C. Chang, One-machine rescheduling heuristics with efficiency and stability as criteria, *Computers and Operations Research*, vol.20, no.1, pp.1-14, 1993.

[20] L. K. Church and R. Uzsoy, Analysis of periodic and event-driven rescheduling policies in dynamic shops, *International Journal of Computer Integrated Manufacturing*, vol.5, no.3, pp.153-163, 1992.

[21] R. J. Abumaizar and J. A. Svestka, Rescheduling job shops under random disruptions, *International Journal of Production Research*, vol.35, no.7, pp.2065-2082, 1997.

[22] L. Tang, W. Liu and J. Liu, A neural network model and algorithm for the hybrid flow shop scheduling problem in a dynamic environment, *Journal of Intelligent Manufacturing*, vol.16, no.3, pp.361-370, 2005.

[23] M. A. Shafia, M. P. Aghaee and A. Jamili, A new mathematical model for the job shop scheduling problem with uncertain processing times, *International Journal of Industrial Engineering Computations*, vol.2, no.2, pp.295-306, 2011.

[24] D. Ouelhadj and S. Petrovic, A survey of dynamic scheduling in manufacturing systems, *Journal of Scheduling*, vol.12, no.4, pp.417-431, 2009.

[25] H. Aytug, M. A. Lawley, K. McKay, S. Mohan and R. Uzsoy, Executing production schedules in the face of uncertainties: A review and some future directions, *European Journal of Operational Research*, vol.161, no.1, pp.86-110, 2005.

[26] G. E. Vieira, J. W. Hermann and E. Lin, Rescheduling manufacturing systems: A framework of strategies, policies and approaches, *Journal of Scheduling*, vol.6, no.1, pp.36-92, 2003.

[27] S. S. Panwalkar and W. Iskander, A survey of scheduling rules, *Operations Research*, vol.25, no.1, pp.45-61, 1977.

[28] H. M. Goldberg, Jackson's conjecture on earliest due date scheduling, *Mathematics of Operations Research*, vol.5, no.3, pp.460-466, 1980.

[29] J. C. Tay and N. B. Ho, Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems, *Computers & Industrial Engineering*, vol.54, no.3, pp.453-473, 2008.

[30] C. D. Geiger and R. Uzsoy, Learning effective dispatching rules for batch processor scheduling, *International Journal of Production Research*, vol.46, no.6, pp.1431-1454, 2008.

[31] T. Hildebrandt, J. Heger and B. Scholz-Reiter, Towards improved dispatching rules for complex shop floor scenarios: A genetic programming approach, *Proc. of the 12th Annual Conf. on Genetic and Evolutionary Computation*, Portland, OR, USA, pp.257-264, 2010.

[32] D. Jakobovic, L. Jelenkovic and L. Budin, Genetic programming heuristics for multiple machine scheduling, *Proc. of the 10th European Conf. on Genetic Programming*, Spain, pp.321-330, 2007.

[33] S. Nguyen, M. Zhang, M. Johnston and K. Tan, A computational study of representations in genetic programming to evolve dispatching rules for the job shop scheduling problem, *IEEE Transactions on Evolutionary Computation*, vol.17, no.5, pp.621-639, 2012.
[34] C. Pickardt, J. Branke, T. Hildebrandt, J. Heger and B. Scholz-Reiter, Generating dispatching rules for semiconductor manufacturing to minimize weighted tardiness, *Proc. of the 2010 Winter Simulation Conf.*, Baltimore, Maryland, USA, pp.2504-2515, 2010.
[35] S. Nguyen, M. Zhang, M. Johnston and K. Tan, Automatic design of scheduling policies for dynamic multi-objective job shop scheduling via cooperative coevolution genetic programming, *IEEE Transactions on Evolutionary Computation*, vol.18, no.2, pp.193-208, 2014.
[36] C. Ferreira, Gene expression programming: A new adaptive algorithm for solving problems, *Complex System*, vol.13, no.2, pp.87-129, 2001.
[37] N. Li, L. Gao, P. Li and X. Li, A GEP-based reactive scheduling policies constructing approach for dynamic flexible job shop scheduling problem with job release dates, *Journal of Intelligent Manufacturing*, vol.24, no.4, pp.763-774, 2013.
[38] C. Raquel and X. Yao, Dynamic multi-objective optimization: A survey of the state-of-the-art, *Evolutionary Computation for Dynamic Optimization Problems*, pp.85-106, 2013.
[39] W. Herroelen and R. Leus, Project scheduling under uncertainty: Survey and research potentials, *European Journal of Operational Research*, vol.165, no.2, pp.289-306, 2005.
[40] L. Jourdan and E. Talbi, Combinatorial optimization of stochastic multi-objective problems: An application to the flow-shop scheduling problem, *Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science*, vol.4403, pp.457-471, 2007.
[41] D. Lei, Simplified multi-objective genetic algorithms for stochastic job shop scheduling, *Applied Soft Computing*, vol.11, no.8, pp.4991-4996, 2011.
[42] X. Zuo, H. Mo and J. Wu, A robust scheduling method based on a multi-objective immune algorithm, *Information Sciences,* vol.179, no.19, pp.3359-3369, 2009.