Received January 26, 2019, accepted February 11, 2019, date of publication February 19, 2019, date of current version March 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2900117

The Internet of Things Enabled Shop Floor Scheduling and Process Control Method Based on Petri Nets

XIAOQIANG WU[1,2], SONGLING TIAN[1,3], AND LEI ZHANG[4]
[1]Key Laboratory of Mechanism Theory and Equipment Design, MOE, Tianjin University, Tianjin 300350, China
[2]College of Mechanical Engineering, Inner Mongolia University for Nationalities, Tongliao 028000, China
[3]School of Mechanical Engineering, Tianjin University, Tianjin 300350, China
[4]School of Mechanical Engineering, Tianjin University of Commerce, Tianjin 300134, China

Corresponding author: Songling Tian (tiansongling1224@sina.cn)

This work was supported in part by the College Scientific Research Project of the Inner Mongolia Autonomous Region under Grant NJZY18159, in part by the Science and Technology Innovation Guide Project of the Inner Mongolia Autonomous Region under Grant KCBJ2018028, in part by the Natural Science Foundation of the Inner Mongolia Autonomous Region of China under Grant 2018MS05025, and in part by the Inner Mongolia Autonomous Region Colleges and Universities' "Young Science and Technology Talents Program" under Grant NJYT-19-B15.

ABSTRACT Shop floor scheduling requires consideration of the dynamic, time-varying, and unpredictable natures of the manufacturing environment. A shop floor scheduling/rescheduling method based on Petri net and ant colony optimization (PN-ACO) is proposed given an abnormal event represented by machine breakdown. Because of the difficulty to schedule in the complex and changeable internal and external environment of the shop floor, an Internet-of-Things (IoT)-enabled process control method is proposed, using sensors, RFID, industrial wireless communication, automatic identification, and other technologies to perceive the shop floor field. The value of the mean relative error is 1.77, which illustrates the feasibility and efficiency of the proposed PN-ACO algorithm to solve flexible job shop scheduling problems. To represent operation sequencing information, a schedule timed transition Petri net (TTPN) is proposed which is evolved from the TTPN. On the basis of the mapping mechanism between the Petri net model and the XML, manufacturing resources become autonomous and interactive distributed intelligent manufacturing resources. The experimental results confirm that the proposed method is effective for scheduling and process control of the IoT-enabled shop floor.

INDEX TERMS Internet of Things enabled shop floor, dynamic scheduling, ant colony optimization, Petri nets, XML, RFID.

I. INTRODUCTION

In the shop floor manufacturing process, studies show that most of the time is spent in the non-processing process. Therefore, an effective production scheduling scheme will help manufacturing enterprises to reduce production costs and improve production efficiency [1]. Manufacturing shop floor scheduling decomposes a given jobs processing task according to certain rules. Under the constraints of production resources, the decomposed sub-tasks are allocated reasonably and the processing sequence is arranged to optimize the time cost to achieve the optimal production efficiency.

The associate editor coordinating the review of this manuscript and approving it for publication was Chunsheng Zhu.

According to the processing state of the jobs, the scheduling problem can be divided into static and dynamic scheduling [2]. Static scheduling refers to all parts that need to be processed and their technological processes are known, which can be used as parameters by scheduling algorithm to calculate all future processing states of the jobs; dynamic scheduling refers to the random processing tasks of the jobs and the real-time dynamic response of scheduling system with various events in the production process, such as machine breakdown, delivery date change, emergency, urgent task, etc. Because of the randomness of various uncertain events in the actual production system, the dynamic scheduling scheme can solve the actual production scheduling more effectively.

2169-3536 © 2019 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

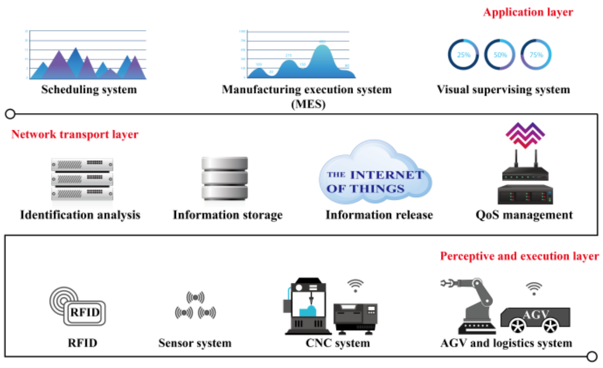


FIGURE 1. The framework of IoMT.

However, the hypothetical conditions in the current scheduling theory research cannot fully describe the actual situation of the shop floor, especially the application of most of the preconditions of the algorithm is narrow, and cannot adapt to the current complex state of the shop floor manufacturing. Taking dynamic scheduling as an example, most of the current algorithms assume that the disturbance information is predictable, which cannot cope well with the complex and random shop floor environment [3]. On the other hand, pre-scheduling plan is often unable to match the real-time status of job changes and various resources such as production equipment and materials in the shop floor, so there are some problems in the execution process such as lag and inconsistency. Faced with the inconsistency between planning and execution, the problem of adaptive control based on shop floor state perception must be solved [4], [5]. Therefore, it is not enough to rely solely on the scheduling system to deal with the complex and changeable internal and external environment of the shop floor. It also needs a higher adaptive and faster response speed control system, namely distributed control system.

In order to cope with the distributed control demand on the shop floor production field, engineers and scholars carry out repeated production reforms from the past. Nowadays, the emergence of the Internet of Things (IoT) has attracted more attention [6].

With the help of IoT technology, we can build an internet of Manufacturing Things (IoMT) environment for shop floor. The shop floor system under IoMT environment can be divided into three basic levels: application layer, network transport layer, and perceptive and execution layer, as shown in Figure 1. Autonomous analysis of system status and real-time response to dynamic events are the main characteristics of shop floor production under IoMT environment. IoMT environment is used to capture real-time information of shop floor production.

The motivation of this paper is to design an IoT enabled shop floor scheduling and process control method based on Petri nets, using sensors, RFID, industrial wireless communication, automatic identification and other technologies to perceive the shop floor field [7].

In our view, the main contributions of this paper include the following:

1) The manufacturing process and resources are analyzed, and the hardware configuration of manufacturing resources is accomplished by modern information technology such as RFID. The real-time state and behavior logic model of manufacturing process and resources based on Petri net are established.

2) Static and dynamic scheduling of Petri nets and heuristic algorithms are researched.

3) The mapping mechanism of Petri net model to XML, the key data needed in manufacturing process, the transformation of Petri net to XML, the manufacturing resources fusion, and the automatic perception and process control of distributed manufacturing resources are separately studied.

This paper is organized as follows. In Section 2, we present the related literature reviews; in Section 3, we describe the shop floor scheduling problem formulation and Petri-net-based representation; section 4 presents the scheduling method for the shop floor environmental production system; section 5 explains the adaptive shop floor control method based on state perception; in Section 6, we present an extensive computational study using several benchmark problems, comparing our results with the state-of-the-art algorithms and conduct experiments on the experimental platform. Some final remarks and future research directions are given in Section 7.

II. RELATED LITERATURE REVIEWS

This section introduces the background of relevant research. We review the general dynamic scheduling methodologies, Petri net and AI heuristic scheduling algorithms.

A. INTERNET OF THINGS (IoT) AND IoT ENABLED SHOP FLOOR

The lack of real-time and accurate information of manufacturing resources and interruptions lead to the delay, distortion and incompleteness of information as well as frequent rescheduling in the shop floor, which are widely recognized by scholars [8], [9].

To address these problems, the IoT-enabled real-time scheduling is proposed for the real-time implementation of the predictive-reactive approach. Real-time information of shop floor production process is sensed and captured based on the constructed IoMT environment [10]. When abnormal events occur, they can be sensed and trigger the reactive scheduling. This predictive-reactive approach can handle the disturbance problem of abnormal events, and minimize the loss of cost.

B. DYNAMIC SCHEDULING METHODOLOGIES

Most research addresses FMS scheduling under a static environment in which no unexpected events occur (i.e., machine breakdown, uncertain processing times, arrival of a new job or cancellation of jobs) during the execution of the schedule [11]. However, real production environments are dynamic and flexible.

Proactive, reactive, and predictive-reactive scheduling approaches are the three main methodologies used to overcome stochastic disruptions. The proactive approach predicts the occurrence of unexpected events, where the initial schedules are generated in anticipation of unexpected events [12]. In the reactive approach, no schedule is generated in advance, but the schedule is modified providing the necessary reactions by priority dispatching rules in real time [13]. The predictive-reactive approach controls and guides recovery of the system from disruptions, where initial schedules are revised to adapt to the new environment caused by dynamic events [14], [15]. Considering the robustness and stability of scheduling results, the predictive-reactive approach is the most commonly used in dynamic scheduling of FMSs (CFJSP or FJSP) [16].

C. PETRI NETS

Petri nets, graph theory and automata are the three main methodologies used to model the FJSP. Petri nets remain the most effective formalism and have been widely studied in the scheduling and control community [17]. Petri net is not only a simple and intuitive graphical modeling tool, but also can simulate the manufacturing system, analyze and evaluate the operational performance of the system (e.g. equipment utilization, reliability, etc.). It can also be used to check and prevent system failures such as machine breakdown, dead lock, resource conflict, etc.

Structural analysis (invariants) and reachability graph analysis are the two main techniques in existence for the analysis of a Petri-net model [18], [19]. The former is typically used in the field of control and FMSs, while the latter is typically used for scheduling work [20]. To simplify the Petri-net model of a manufacturing system, certain sub-net approaches have been presented, such as the PPN [21], S^3PR [22] and S^4PR [23]. In addition, a colored Petri net (CPN) can reduce the total number of places in a model [24].

Petri net model is a graphical description of process data in manufacturing system. It does not have the ability of storage itself, that is, it is an abstract model that cannot be directly used in intelligent manufacturing system to guide the manufacturing process. In order to apply the model in manufacturing system and combine Petri net with industrial interconnection technology, storable conversion of Petri nets is needed [25]. A more advanced feature is that the visualized Petri net model can be transformed into a stored language and stored in the RFID tag to be equipped with terminal perception and analysis, and then intelligent decision-making can be made on the manufacturing process.

D. AI HEURISTIC SCHEDULING ALGORITHMS

The recent integration of Petri nets as a representation tool with AI heuristic and metaheuristic methods as a reasoning paradigm appears to be a promising answer to the need to develop models that incorporate the full complexity of the FMS (CFJSP or FJSP) yet are efficient enough to obtain satisfactory solutions [26]. Ahmad and Khan [27] presented a method for a periodic JSP by combining a timed place Petri net (TTPN) with the heuristic search method. Baruwa and Piera [28] presented a simulation optimization approach employing an anytime search method to optimize FMS scheduling problems modelled using timed coloured Petri net (TCPN) formalism. Caballero-Villalobos *et al.* [29] addressed complex production scheduling problems using Petri nets and genetic algorithms (GAs).

A metaheuristics is a higher-level heuristic design procedure that aims to find, generate, or select a heuristics that may provide a satisfactory solution to the FJSP [30]. Many metaheuristics implement some form of stochastic optimization and can find acceptable solutions with less computational effort than traditional optimization algorithms, simple heuristics, or iterative methods [31].

One class of metaheuristics algorithms known as evolutionary metaheuristics (e.g., evolutionary computation and ant colony optimization, ACO) incorporates a learning component in the sense that these algorithms try to learn correlations among decision variables, aiming to identify high-quality areas in the search space. In ACO, a near-optimal solution can be obtained iteratively by a biased sampling of the search space, i.e., identifying high-quality areas according to a probability distribution [32].

III. PROBLEM FORMULATION AND PETRI-NET-BASED REPRESENTATION

The CFJSP, where there may be a multi-batch for each job, is different with the traditional FJSP. The multi-batch problem, however, can be deconstructed into independent jobs. The CFJSP can be transformed into an FJSP. As deconstruction attempts increase the scale of the problem, it is necessary to propose a suitable approach for modeling the problem of large-scale complex system scheduling, which will be elaborated in this section.

A. PROBLEM FORMULATION

The FJSP, also called the general job-shop problem with parallel (or alternative) resources, can be formulated as follows. Let $J = \{1, 2, \ldots, n\}$ be a set of jobs, each of which constitutes a predetermined sequence of operations. Each operation O_{ij} (operation j of job i) is to be processed on one machine M_{ij} from the set of eligible machines $M = \{1, 2, \ldots, m\}$. We assume that the processing time P_{ijk} of the operations O_{ij} on machine $M_k (M_k \in M_{ij})$ is known and that each machine can process only one operation at a time.

The problem consists of the job allocation to machines of an adequate type and the schedule of job execution determination on each machine to minimize the makespan (or the maximum completion time of all jobs) [33].

B. TIMED PETRI NETS

The FJSP arises when multiple job types are processed respectively by multiple types of shared resources according to their production process and constraints [34]. Petri nets,

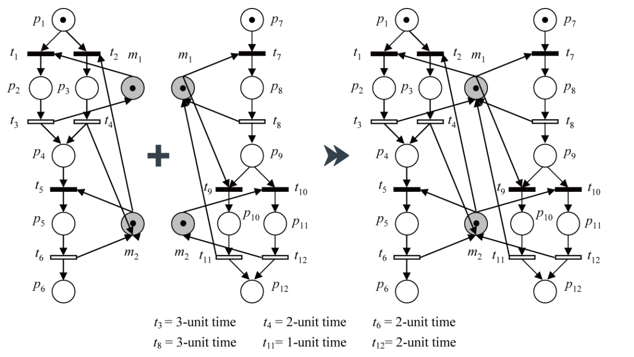


FIGURE 2. **The TTPN model of example 1.**

```
Procedure: Reachability graph ( )
Input: Current simulation clock $t$, current marking $m_k$, goal marking $m_{goal}$.
Output: A firing sequence of the transitions.
// Initialization
Reading the Petri net data; iteration number $\leftarrow k$
// Main loop
while $m_k \neq m_{goal}$
    for $t = t$ to $t_{end}$
        Calculating each transition;
        Constructing the set of enabled transitions $\tau$ at current marking $m_k$;
        if $\tau \neq \varnothing$
            Selecting a transition $\tau'$ of $\tau$, firing $\tau'$ and calculating the successor marking $m_{k+1}$;
            Recording $\tau'$ and $m_{k+1}$, $k \leftarrow k+1$, $m_k \leftarrow m_{k+1}$;
        else
            $t = t+1$;
        end
    end
end
```

FIGURE 3. **Expanding procedure of the reachability graph.**

as a modeling and analysis tool, can describe the production process and constraints in a graphical way. Furthermore, timed Petri nets can provide information about the FJSP in the form of mathematical analysis tools and reachability graphs that can be used to guide the scheduling process [35]. Thus, timed Petri nets are an ideal tool for modeling the FJSP [36].

Generally, there are two types of timed Petri nets: timed placed Petri nets (TPPNs), which associate time to places, and TTPNs, which associate time to transitions [37]. There is an inevitable drawback: the model size of TPPNs is much larger than that produced by TTPN; thus, we select the TTPN to model the FJSP, meaning that time is associated only with transitions and that all place firings are instantaneous.

C. MODELLING AN FJSP WITH TTPNs

Modeling methods for the FJSP are often divided into three types based on Petri nets: bottom-up modeling, top-down modeling, and hybrid modeling. Bottom-up modeling approaches first obtain the Petri nets for each job and then obtain the final model by place or transition fusion. In contrast, the top-down modeling approach first expresses the overall problem macroscopically and then gradually refines each job until the problem is fully described clearly. The hybrid modeling approach integrates top-down modeling for the operation and bottom-up modeling for the resources.

Our proposed hybrid modeling method follows the concept of S^3PR, which is a hybrid modeling approach for discrete event systems [22]. One example (Example 1) of this modeling method is illustrated in Figure 2. In this example, the first operation ($O1$) of $job1$ can be operated on machine $M1$ for 3 units of time or on machine $M2$ for 2 units of time; the second operation ($O2$) of $job1$ can be operated on machine $M2$ for 2 units of time; the third operation ($O3$) of $job2$ can be operated on machine $M1$ for 3 units of time; the forth operation ($O4$) of $job2$ can be operated on machine $M1$ for 1 unit of time or on machine $M2$ for 2 units of time. Places p_1 and p_7 are the start places of the respective jobs, and p_6 and p_{12} are the finish places. $t_1, t_2, \ldots, t_{12}$ represent the relative operations. p_4 and p_9 are buffer places between operations. M_1 (m_1) and $M_2(m_2)$ are shared machines for these operations.

Once the Petri net model is constructed, given the current and goal marking, we can use the ACO algorithm to obtain the optimal rescheduling.

IV. SCHEDULING METHOD OF THE FJSP

Theoretically, an optimal schedule can be obtained by generating a reachability graph and finding the optimal path from the initial marking to the goal marking. The path is a firing sequence of the transitions of the TTPN model [26]. Even for a simple Petri net, however, a reachability graph may be too large to generate in its entirety. Instead of generating the entire reachability graph, a heuristic search algorithm is employed, which can be expended as the loop iteration procedure in Figure 3.

A. HEURISTIC FUNCTIONS

The scheduling algorithm presented here formulates a scheduling problem using a TTPN model, employs a heuristic search, and limits the search space by the use of an ACO-based metaheuristics search algorithm. An ant constructively builds a solution to the problem by moving through nodes of the construction reachability graph. Ants move by applying a stochastic local decision policy that makes use of the pheromone values and the heuristic values on the connections of the construction reachability graph.

The ant chooses an unsearched feasible marking j among the set of unsearched feasible nodes (or feasible markings neighborhood N) by the probability, and a probability distribution given by

$$P_{ij}^{l} = \begin{cases} \dfrac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{i\in N_i}\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}, & j \notin N_i^l \\ 0, & j \notin N_i^l \end{cases} \tag{1}$$

where τ_{ij} is the pheromone trail associated with the movement from node (marking) i to batch node (marking) j, η_{ij} is the heuristic desirability of moving node (marking) i to node (marking) j, N_i^l is the neighborhood at node (marking) i that is searched by ant l, and α and β are parameters that determines the relative importance of the heuristic information.

Definition 1: The function γ_{ij}, which is the actual completion time required from node (marking)M_i to one-step reachable node (marking) M_j, is obtained as follows:

$$\gamma_{ij} = Et\,(t) + Dt\,(t) \tag{2}$$

where marking M_j can be reached through firing transition t, $Et(t)$ is the real response time that transition t is enabling, and $Dt(t)$ is the time delay associated with the transition t.

The heuristic function η_{ij} used in this study is adapted from the admissible heuristic δ_{ij} proposed by Reyes *et al.* [38]. The resource cost reachability (RCR) matrix is the kernel of the heuristic function proposed by Reyes, Yu, Kelleher, & Lloyd, which represents the minimum machine utilization for processing a subpart in an ideal no-block situation.

Definition 2: The function δ_{ij}, which estimates the remaining time necessary from the one-step reachable node (marking) M_j to the final destination node (marking) M_f, is obtained as follows:

$$\delta_{ij} = \frac{\sum_{k=1}^{n} RCR\left(P_{k_start}, P_{k\_end}\right)}{m} \tag{3}$$

where P_{k_start} represents the indexes of nonzero digit to marking M_j, $RCR\ (P_{k\_\text{start}}, P_{k\_\text{end}})$ represents the minimum machine utilization for processing a product of the kth job, m is the number of resources (machines) in the system, and n is the number of jobs.

Definition 3: Using the above γ_{ij} and δ_{ij}, we define the heuristic function η_{ij} for ACO as follows:

$$\eta_{ij} = \gamma_{ij} + \delta_{ij} \tag{4}$$

Proposition 1 ($\boldsymbol{\eta_{ij}}$ is Admissible): By the γ_{ij} definition, for a given marking, there is a transition sequence of the path $O_{i\_end} = tO_{j\_end}$ (i.e., $M_i\ [t > M_j$ and $M_j\ [O_{j\_end} > M_f$), so that makespan is the minimum, where marking M_j can be reached from marking M_i through firing transition t, and O_{i_end} represents the transition sequence of the path from marking M_i to final destination marking M_f. Thus, we can express $\sum Dt(O_{i\_end})$ for the time delay or makespan in this situation.

Let us consider an alternate transition sequence $O_{i\_end}*$ for the marking M_i. By Definition 1 and Definition 2, $\gamma_{ij} \leq \gamma_{ij}*$, $\delta_{ij} \leq \delta_{ij}*$. Thus, $\eta_{ij} \leq \eta_{ij}*$, and $\sum Dt(O_{i\_end}) \leq \sum Dt(O_{i\_end})*$. This result implies that for any other possibility sequence $O_{i\_end}*$, the makespan will be equal to or greater than O_{i_end}.

B. SOLUTION PROCEDURE OF STATIC SCHEDULING

While moving, the ant keeps in memory the partial solution it has built in terms of the path it was following on the construction reachability graph. Once the goal marking is found, the optimal path will be constructed by tracing the pointers that denote the parenthood of the markings, from the goal marking to the initial marking. Then, the transition sequence of the path provides the order of the initiation of the activities, i.e., the scheduling result.

```
Procedure: PN-ACO based metaheuristic algorithm
Inputs: Given an FJSP, we generate the TTPN model using the method proposed in section 3.
Output: Getting an optimal feasible schedule.
    Step 1 Initializing algorithm.
    Step 1.1 Reading and storing the Petri net data corresponding to the scheduling problem.
    Step 1.2 Initializing data values of simulation, initialize $\tau_{ij}$ to 1.
    Step 1.3 Constructing and initializing the data storage structure.
    //Main loop
    //Step 2 Ant searching loop
    In the Ant loop, we put $m$ ants at the top (or initial marking) of the reachability graph. Each
of the $m$ ants constructs a sequence of distinct nodes (markings) which is a branch of the
reachability graph.
    Step 2.1 Calculating the function $\tau_{ij}$ and $\eta_{ij}$ for each one-step reachable nodes.
    Step 2.2 Constructing a feasible solution by each ant.
    The ant chooses an unsearched feasible marking with the least evaluation function $P^{l}{}_{ij}(\bullet)$.
    Step 2.2 Local search
    The local search method used in PN-ACO algorithm is the roulette wheel selection (RWS).
At the same time, the node shift must follow the Enabling Rule and Firing Rule of Petri nets, else
promote simulation clock and turn to step 2.1.
    Step 2.3 Global update of pheromone trail
    Following the Formula (5), the global updating rule is applied after each ant has completed a
feasible solution.
    End Ant searching loop
    Following the Formula (6), the pheromone trail is added to the path of the incumbent global
best solutions.
    End Main loop
End Procedure
```

FIGURE 4. **Procedure of PN-ACO-based metaheuristics algorithm.**

If the path of node (marking) i to node (marking) j belongs to one part of incumbent global best solution BPath, then

$$\Delta\tau_{ij}^{l} = \begin{cases} \dfrac{Q}{L^{l}}, & f(i,j) \in \text{BPath} \\ 0, & f(i,j) \notin \text{BPath} \end{cases} \tag{5}$$

where Q is a constant and L is the time delay from current node (marking) i to node (marking) j.

To avoid premature convergence, the update reduces the amount of pheromone for the newly added node (marking) to discourage the following ants from choosing the same batch. This step is achieved using the following global updating rule:

$$\tau_{ij}(t+1) = (1-\rho)\cdot\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{l}(t) \tag{6}$$

where ρ $(0 < \rho \leq 1)$ is a parameter representing the evaporation of pheromone.

The proposed Petri-net- and ACO-based metaheuristic algorithm (PN-ACO-based metaheuristic algorithm) is illustrated in Figure 4.

C. DYNAMIC SCHEDULING OF THE FJSP

In practice, the operations of the FJSP may be interrupted and delayed due to uncertain or stochastic factors. Machine breakdown, which is a common factor of uncertainty, corresponds to a period of time when a certain machine is not available to process operations.

Machine breakdown is an inherent uncertainty imposed on static scheduling (or predictive scheduling). When this occurrence happens, rescheduling the decision (or reactive scheduling) may be necessary, i.e., if the disruption effect cannot be absorbed by the predictive schedule, then a new

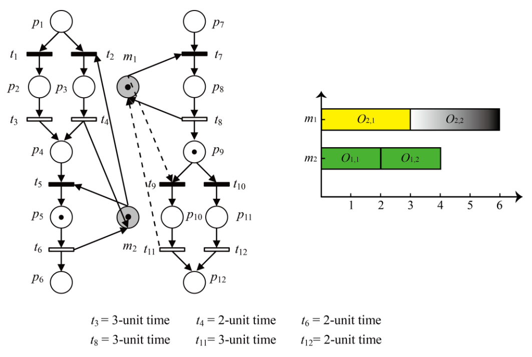


FIGURE 5. A TTPN model for a dynamic FJSP.

reactive schedule is generated considering that the jobs have not been processed, or the predictive scheduling is kept.

If the disruption effect cannot be absorbed by the predictive schedule, operations waiting for scheduling will have to either: (1) delay their processing (i.e., right-shift scheduling (RSS) strategies), or (2) be reassigned to another machine (i.e., routing changing scheduling (RCS) strategies).

In this study, the following assumptions are made about the nature of interruptions.

1) The end time of an interruption is known at the time it starts.

2) Operations are not resumable. If an operation is affected by an interruption, the operation must restart its processing anew. The work of the affected operation must be redone from its start.

Figure 5 shows the TTPN model of a dynamic FJSP, which is an FJSP (Example 1) experiencing a breakdown scenario in which machine 1 at time 3 needs 1 unit of time to be recovered.

When machine breakdown occurs, in this method, we need only modify the time delays associated with transitions rather than completely remodeling. Figure 6 shows the modify method and the reactive scheduling strategy: (a) shows the RCS strategy, and (b) shows the RSS strategy.

After the establishment of the model of the dynamic flexible job-shop problem with random machine breakdowns, in this study, a PN-ACO-based predictive-reactive algorithm (a three-step optimum scheduling method) is presented for predictive-reactive scheduling policy to achieve a more stable and robust solution. The first step generates a predictive schedule, which can use the mentioned PN-ACO algorithm in Section 3. The second step identifies the affected operations and modifies the Petri nets using the approach diagrammed in Figure 4. The third step finds the optimal scheduling result by searching a reachability graph with ACO. This reactive schedule is built using the same scheduling algorithm proposed for generating a predictive schedule, i.e., PN-ACO, but in this case with a smaller number of jobs. The PN-ACO-based predictive-reactive algorithm is shown in Figure 7.

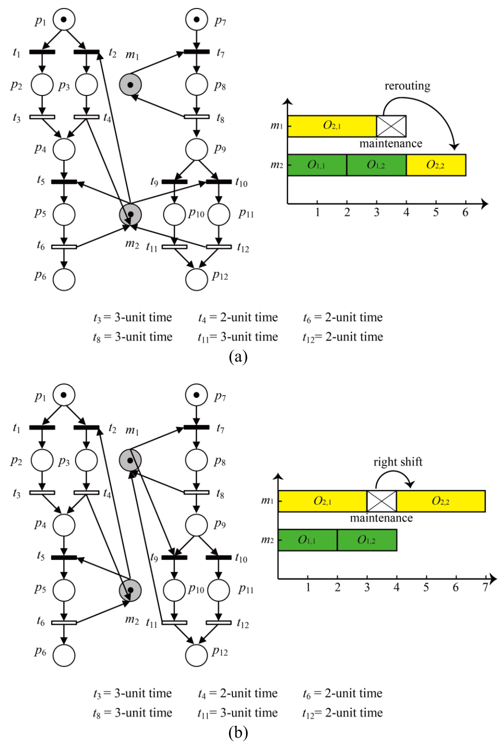


FIGURE 6. (a) RCS strategy for solving a dynamic FJSP. (b) RSS strategy for solving a dynamic FJSP.

V. ADAPTIVE SHOP FLOOR CONTROL BASED ON STATE PERCEPTION

This section mainly designs all kinds of data information needed by the Petri net model in the manufacturing process, and studies the mapping mechanism between the Petri net model and the XML statements that can be stored. On this basis, these data, together with the Petri net model, are transformed into the integration of XML and manufacturing resources, so that manufacturing resources can become autonomous and interactive distributed intelligent manufacturing resources.

A. STORABLE CONVERSION OF PETRI NET MODEL BASED ON XML

To represent operation sequencing information, we present the concept of a schedule timed transition Petri net (STTPN) that is evolved from the TTPN. Unlike a TTPN, in which time delays are associated with transitions, the STTPN associates the time of firing to transitions.

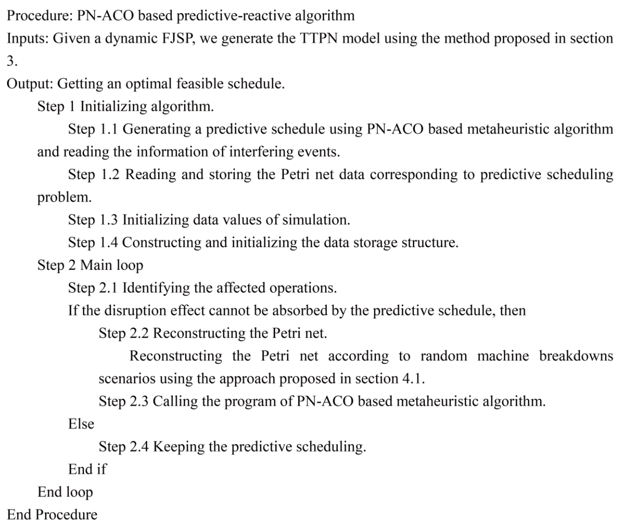
Definition 4: An STTPN is a six-tuple $Z = \{P, T, I, O, M_0, F_t\}$, where

1) $P = \{p_1, p_2, \ldots, p_m\}$ is a finite set of places;

2) $T = \{t_1, t_2, \ldots, t_n\}$ is a finite set of transitions with $P \cup T \neq \emptyset$ and $P \cap T = \emptyset$;

3) $I: P \times T \Rightarrow N^{+} \cup \{0\}$ is an input function that defines directed arcs from places to transitions;

4) $O: T \times P \Rightarrow N^{+} \cup \{0\}$ is an output function that defines directed arcs from transitions to places;



```
Procedure: PN-ACO based predictive-reactive algorithm
Inputs: Given a dynamic FJSP, we generate the TTPN model using the method proposed in section
3.
Output: Getting an optimal feasible schedule.
    Step 1 Initializing algorithm.
        Step 1.1 Generating a predictive schedule using PN-ACO based metaheuristic algorithm
    and reading the information of interfering events.
        Step 1.2 Reading and storing the Petri net data corresponding to predictive scheduling
    problem.
        Step 1.3 Initializing data values of simulation.
        Step 1.4 Constructing and initializing the data storage structure.
    Step 2 Main loop
        Step 2.1 Identifying the affected operations.
        If the disruption effect cannot be absorbed by the predictive schedule, then
            Step 2.2 Reconstructing the Petri net.
                Reconstructing the Petri net according to random machine breakdowns
            scenarios using the approach proposed in section 4.1.
            Step 2.3 Calling the program of PN-ACO based metaheuristic algorithm.
        Else
            Step 2.4 Keeping the predictive scheduling.
        End if
    End loop
End Procedure
```

FIGURE 7. Expanding procedure of the PN-ACO-based dynamic scheduling algorithm.

5) M_0: $P \Rightarrow N^+ \cup \{0\}$ is a function that defines the initial marking;

6) $F_t = \{F_{t1}, F_{t2}, \ldots, F_{tn}\}$ is a finite set of firing times associated with transitions.

The key in the realization of the IoT is the RFID technology. The connection of things in the IoT enabled shop floor is realized by the technology of RFID. STTPN can be used to express the scheduling scheme and the operation state of the shop floor, but STTPN is a graphical description of the state of manufacturing resources and the logic of behavior, which is not easy to access the model. Therefore, STTPN model needs to be stored in the RFID tag of the perceived job through reading and writing. Petri Nets Markup Language (PNML) can be used to transform Petri nets model and realize STTPN model storage transformation and standardization.

The meta-model defines the basic structure of PNML file. Figure 8 shows the component objects of the meta-model and the relationship among them.

B. STORABLE CONVERSION OF PETRI NET MODEL BASED ON XML

The mapping and storage transformation of Petri net model can make the state and behavior logic of resources represented by Petri net be read and parsed by control terminals, and then realize the automatic perception and process control of distributed manufacturing resources.

In order to present the state of the job in an all-round way, the job model based on Petri net should also give the comprehensive information of the job, including the number of equipment used in job processing and handling, the number of job processing process, the number of moving path, the start and end time of each operation, the qualified results of key quality testing, the planned processing time, etc. Because the model itself is an abstract description of the job information, which is omitted in the job model, it needs to be embedded

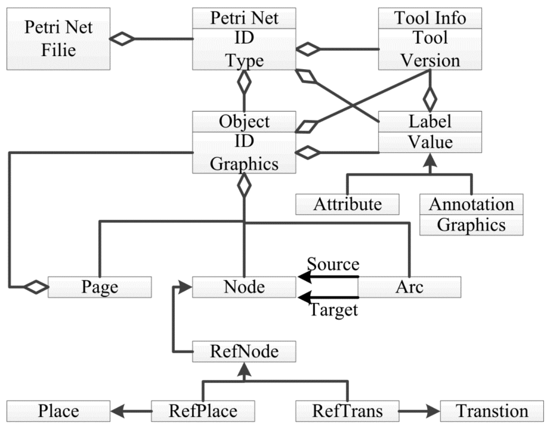
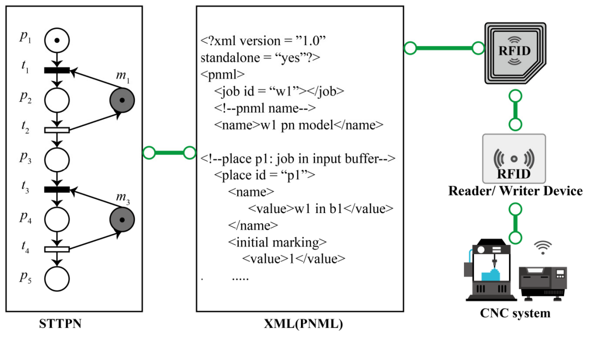


FIGURE 8. The basic structure of PNML meta model.



```
<?xml version = "1.0"
standalone = "yes"?>
<pnml>
   <job id = "w1"></job>
   <!--pnml name-->
   <name>w1 pn model</name>

<!--place p1: job in input buffer-->
   <place id = "p1">
      <name>
         <value>w1 in b1</value>
      </name>
      <initial marking>
         <value>1</value>
.        .....
```

FIGURE 9. The diagram of initialization process of manufacture.

when the Petri net model is transformed into the storage language.

According to the writing time of data, the data in the job RFID tag can be divided into two categories: pre-set data and record data. Pre-set data is the data written into the RFID tag through initialization before the job goes online. Recorded data is the data written into the RFID tag in real time during the manufacturing process.

In the process of production initialization, the system determines the product type according to the production plan, assuming w, automatically generates the job number w_1, obtains the process route of the job, and maps the state, activity and behavior logic of the job in the manufacturing process to the corresponding Petri net model according to the relationship between the operations in the process route.

As shown in Figure 9, the XML file of information such as numbering, process number and quality standard is imported into the RFID tag attached to the job. For the manufacturing equipment resources to be used, the Petri net model of its state and activity is also generated and mapped to the XML file to be stored in the basic database. These XML files are parsed in the manufacturing process for visual resource awareness and process control.

When the job w_1 reaches the input buffer of the processing machine m_1, the m_1 first judges whether to process the job according to its idle or non-idle state. If it is in the non-idle state, the job will wait in the input buffer. If it is in the idle state, the job will be processed. The detailed procedure is given as follows.

Step 1: The processing machine m_1 reads the PNML file in $_1$ tag through its own RFID reader, and parses the status of the job in the Petri net model, the job ID, the process number and other information in its terminal controller. If the job w_1 is not in the corresponding state to be processed, it will send an abnormal alarm. Otherwise, it will go to step 2.

Step 2: According to the parsed information, the corresponding processing technology is matched in the terminal control machine of the processing equipment, and the processing machine processes the job w_1 according to this technology; the token in the job w_1's process place P_1 is deleted, the transition T_1 is fired, the token in the processing machine place is deleted, and the token flows to the buffer place P_2 (indicating that the transfer equipment is in the transfer process); the processing time will start, the device number and other information are written into the PNML file of the job tag by the reader, and the status of the PNML file in the job tag is updated by the writer (i.e., the number of tokens in the corresponding place of the PNML file is reduced by 1).

Step 3: When the processing is finished, the reader reads the tag information of the buffer device. The terminal controller of the processing machine parses and judges the collected information. If the job w_1 in the buffer device is in the non-idle state, the job will wait in the processing machine. If it is in the idle state, the procedure will go to step 4.

Step 4: Processing machine triggers the transition of completion processing. The job w_1 is put into output buffer. Job quality information, end processing time, processing status and other information are written into job tag.

VI. COMPUTATIONAL EXPERIMENTS

As mentioned, after a modification of Petri nets is conducted, the dynamic FJSP with random machine breakdowns is simplified to solve the classical FJSP. To test the performance of the proposed PN-ACO algorithm and PN-ACO-based predictive-reactive algorithm, extensive experimental evaluation and comparisons with existing algorithms are provided using well-known FJSP benchmark sets.

In what follows, we discuss how to implement static scheduling, dynamic scheduling and supervisory control of an FMS, specifically, a 2 × 3 flexible job-shop problem of FMS.

A. EXPERIMENT ON BENCHMARK INSTANCES PROBLEM

The PN-ACO-based metaheuristics algorithm described above was implemented in MATLAB (The MathWorks Inc., www.mathworks.com) on a PC operating at 2.0 GHz with 2.0 GB of RAM and tested on a large amount of problem instances from the literature. The best result (best makespan) was selected for several sets of benchmark problem instances after five runs. In addition, we reported a comparison between our PN-ACO-based metaheuristics algorithm and other famous algorithms in terms of the makespan. We considered the benchmark instances problem of DP data, which consists of 18 test problems from Dauzère-Pérès and Paulli [39].

TABLE 1. Performance comparison between PN-ACO and the state-of-the-art algorithms on DP data.

Instan-ce	*LB*	TS	hGA	CDDS	GA+TS	PN-ACO	
		C_M	C_M	C_M	C_M	C_M	MRE (%)
01a	2505	2518	2518	2518	2505	*2505	0.00
02a	2228	2231	2231	2231	2230	*2230	0.09
03a	2228	2229	2229	2229	2229	*2228	0.00
04a	2503	2503	2515	2503	2503	*2503	0.00
05a	2189	2216	2217	2216	2212	*2212	1.05
06a	2162	2203	2196	2196	2197	*2187	1.16
07a	2187	2283	2307	2283	2279	*2279	4.21
08a	2061	2069	2073	2069	2067	*2067	0.29
09a	2061	2066	2066	2066	2065	*2065	0.19
10a	2178	2291	2315	2291	2287	2288	5.05
11a	2017	2063	2071	2063	2060	*2060	2.13
12a	1969	2034	2030	2031	2027	*2027	2.95
13a	2161	2260	2257	2257	2248	2250	4.12
14a	2161	2167	2167	2167	2167	*2164	0.14
15a	2161	2167	2165	2165	2163	*2163	0.09
16a	2148	2255	2256	2256	2249	2250	4.75
17a	2088	2141	2140	2140	2140	*2137	2.35
18a	2057	2137	2127	2127	2132	*2124	3.26
MRE (%)	/	2.01	2.12	1.94	1.82	1.77	

Table 1 compares our PN-ACO to the tabu search method (TS) of Mastrolilli and Gambardella [40], the genetic algorithm of Gao *et al.* [41], the climbing depth-bound discrepancy search algorithm (CDDS) of Ben Hmida *et al.* [42], and the hybrid genetic and tabu search algorithm (GA+TS) of Li and Gao [43] on DP data.

The first columns indicate the instance name. In the third column, LB denotes the optimum makespan if known, otherwise, it is the best lower and upper bound so far. The fourth column refers to our best makespan over five runs of PN-ACO, together with the mean relative error (MRE) to the LB. The MRE is calculated as follows:

$$MRE = (C_M - LB)/LB \times 100\% \tag{7}$$

where C_M is the best makespan obtained by PN-ACO and LB is the best-known lower bound.

The remaining columns represent the best results of the four state-of-the-art algorithms in this comparison. In Figure 10, we show the Gantt chart for the 03a test problem. (The parameter settings of PN-ACO are as follows: $\alpha = 1.0$, $\beta = 0.5$, $\rho = 0.05$, and $Q = 2$).

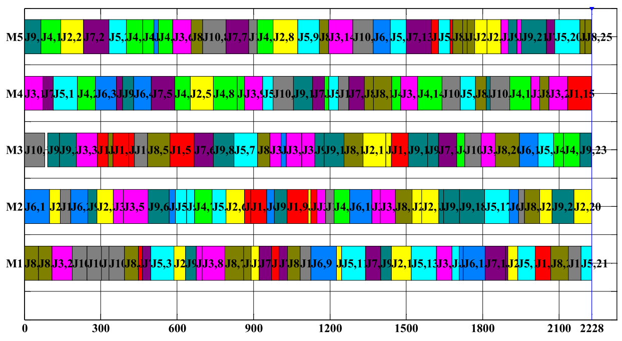


FIGURE 10. Gantt chart of instance 03a in the experiment involving DP data.

TABLE 2. Processing time of each job.

Operations	Machines M_1 (CNC lathe)	M_2 (Machining centre)	M_3 (Machining centre)
O_{11} - Part1	2.5	∞	∞
O_{12}- Part1	∞	8	10
O_{21}- Part2	2.5	∞	∞
O_{22}- Part2	∞	8	10

B. APPLICATION TO AN ENGINEERING PROBLEM

We consider a simple 2 × 3 FJSP to observe the effectiveness of our PN-ACO in solving the engineering problem. In a 2 × 3 FJSP of defence and security facilities, there are three machines and three jobs, and the batch of production is 5 and 5, respectively.

The process times of jobs in each machine are shown in Table , where rows correspond to operations and columns correspond to machines. In this table, the entries of the input table are the processing times, and symbol ∞ means that a machine cannot execute the corresponding operation, i.e., this machine does not belong to the subset of compatible machines for that operation.

The corresponding S^{3}PR is shown in Figure 11. The parameter settings of PN-ACO are as follows: $\alpha = 1.0$, $\beta = 0.5$, $\rho = 0.05$, and $Q = 2$.

In Figure 12 (a), we show the Gantt chart for the FMS scheduling problem, which is the result of predictive scheduling. Then, we consider the effect of machine breakdowns with the FJSP. In this work, we first represent a machine breakdown scenario by L (2, 10, 5), which indicates that machine 2 at time 10 needs 5 units of time to be recovered. By use of the predictive-reactive algorithm proposed in Section 4 and the adaptive shop floor control method based on state perception, the Gantt chart of rescheduling is shown in Figure 13 (b).

C. RESULTS DISCUSSION

In this paper, the average relative error in the optimal solution of 18 problems is 1.77, which is less than that of the other contrast algorithms, fully illustrating the efficiency of the scheduling and rescheduling algorithm PN-ACO for FJSP.

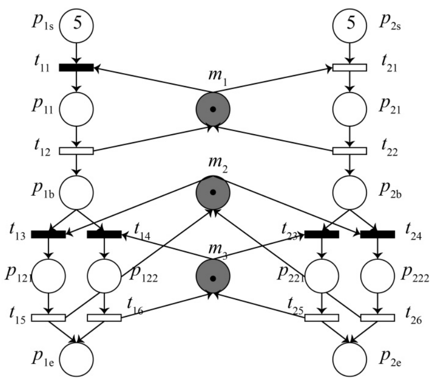


FIGURE 11. Diagram of S^3 PR modeling.

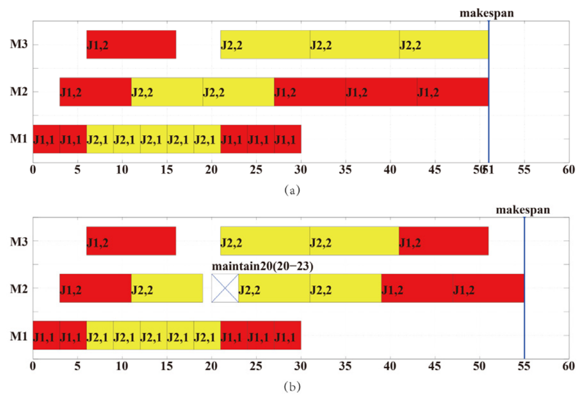


FIGURE 12. (a) Gantt chart of the static scheduling. (b) Gantt chart of the dynamic scheduling with machine breakdown.

From the results of the above engineering experiments, it can be concluded that the control process of the IoT based manufacturing shop floor is completed on the basis of state perception among manufacturing equipment. The Petri nets based information interaction and process control method can be used to solve the dynamic scheduling problems according to the real-time status of the IoT enabled shop floor.

VII. CONCLUSION AND PROSPECTS

This paper offers a shop floor scheduling and process control based on Petri nets. The application of industrial IoT technology can reasonably assign and manage the resources of orders, materials and equipment, which makes the manufactory enterprise have the ability of dynamic shop floor scheduling. A shop floor scheduling/rescheduling method based on Petri nets and ant colony optimization (PN-ACO) is proposed which combines the scheduling algorithm with the predictive-reactive dynamic scheduling strategy. In this paper, the average relative error in the optimal solution for 18 benchmark instances problems from DP data is 1.77, which demonstrates that the proposed PN-ACO algorithm is feasible and has better performance than the comparison state-of-the-art algorithms. An IoT enabled process control method is proposed. The feasibility and validity of the overall technical scheme and related technologies of scheduling and control approach based on the Petri nets are verified by IoT-enabled shop floor field test and test run.

Further work will consider the multi-objective scheduling and using the proposed method to study the dynamic FJSP under other abnormal event interruptions.

REFERENCES

[1] W. Zhang, Y. Yang, S. Zhang, D. Yu, and Y. Chen, "A new three-dimensional manufacturing service composition method under various structures using improved Flower Pollination Algorithm," *Enterprise Inf. Syst.*, vol. 12, no. 5, pp. 620–637, 2018.

[2] L. Ribeiro, A. Rocha, A. Veiga, and J. Barata, "Collaborative routing of products using a self-organizing mechatronic agent framework—A simulation study," *Comput. Ind.*, vol. 68, pp. 27–39, Apr. 2015.

[3] Y. Feng, Q. Wang, Y. Gao, J. Cheng, and J. Tan, "Energy-efficient job-shop dynamic scheduling system based on the cyber-physical energy-monitoring system," *IEEE Access*, vol. 6, pp. 52238–52247, 2018.

[4] S. Răileanu, F. Anton, T. Borangiu, S. Anton, and M. Nicolae, "A cloud-based manufacturing control system with data integration from multiple autonomous agents," *Comput. Ind.*, vol. 102, pp. 50–61, Nov. 2018.

[5] S. Liu, W. Bai, G. Liu, W. Li, and H. M. Srivastava, "Parallel fractal compression method for big video data," *Complexity*, vol. 2018, pp. 1–16, 2018.

[6] N. Moustafa, E. Adi, B. Turnbull, and J. Hu, "A new threat intelligence scheme for safeguarding industry 4.0 systems," *IEEE Access*, vol. 6, pp. 32910–32924, 2018.

[7] T. M. Fernández-Caramés and P. Fraga-Lamas, "A review on human-centered IoT-connected smart labels for the industry 4.0," *IEEE Access*, vol. 6, pp. 25939–25957, 2018.

[8] L. Chen, C. Roberts, F. Schmid, and E. Stewart, "Modeling and solving real-time train rescheduling problems in railway bottleneck sections," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 1896–1904, Aug. 2015.

[9] Y. Zhang, J. Wang, S. Liu, and C. Qian, "Game theory based real-time shop floor scheduling strategy and method for cloud manufacturing," *Int. J. Intell. Syst.*, vol. 32, no. 4, pp. 437–463, 2017.

[10] Y. Zhang *et al.*, "The 'Internet of Things' enabled real-time scheduling for remanufacturing of automobile engines," *J. Cleaner Prod.*, vol. 185, pp. 562–575, Jun. 2018.

[11] D. Rahmani and R. Ramezanian, "A stable reactive approach in dynamic flexible flow shop scheduling with unexpected disruptions: A case study," *Comput. Ind. Eng.*, vol. 98, pp. 360–372, Aug. 2016.

[12] S. Goren, I. Sabuncuoglu, and U. Koc, "Optimization of schedule stability and efficiency under processing time variability and random machine breakdowns in a job shop environment," *Nav. Res. Logistics*, vol. 59, no. 1, pp. 26–38, 2012.

[13] N. Liu, M. A. Abdelrahman, and S. Ramaswamy, "A complete multiagent framework for robust and adaptable dynamic job shop scheduling," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 37, no. 5, pp. 904–916, Sep. 2007.

[14] X.-N. Shen and X. Yao, "Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems," *Inf. Sci.*, vol. 298, no. 219, pp. 198–224, Mar. 2015.

[15] S. Liu, W. Fu, L. He, J. Zhou, and M. Ma, "Distribution of primary additional errors in fractal encoding method," *Multimedia Tools Appl.*, vol. 76, no. 4, pp. 5787–5802, 2017.

[16] D. Sun, W. He, L.-J. Zheng, and X.-Y. Liao, "Scheduling flexible job shop problem subject to machine breakdown with game theory," *Int. J. Prod. Res.*, vol. 52, no. 13, pp. 3858–3876, 2014.

[17] C. A. Petri, "Communication with automata," *Appl. Data Res.*, vol. 15, no. 8, pp. 357–362, 1966.

[18] M. Uzam, Z. Li, G. Gelen, and R. S. Zakariyya, "A divide-and-conquer-method for the synthesis of liveness enforcing supervisors for flexible manufacturing systems," *J. Intell. Manuf.*, vol. 27, no. 5, pp. 1111–1129, Oct. 2016.

[19] M. Bashir *et al.*, "A minimal supervisory structure to optimally enforce liveness on Petri net models for flexible manufacturing systems," *IEEE Access*, vol. 5, pp. 15731–15749, 2017.

[20] Y. Chen, Z. Li, M. Khalgui, and O. Mosbahi, "Design of a maximally permissive liveness- enforcing Petri net supervisor for flexible manufacturing systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 8, no. 2, pp. 374–393, Apr. 2011.

[21] Z. A. Banaszak and B. H. Krogh, "Deadlock avoidance in flexible manufacturing systems with concurrently competing process flows," *IEEE Trans. Robot. Autom.*, vol. 6, no. 6, pp. 724–734, Dec. 1990.

[22] J. Ezpeleta, J. M. Colom, and J. Martínez, "A Petri net based deadlock prevention policy for flexible manufacturing systems," *IEEE Trans. Robot. Autom.*, vol. 11, no. 2, pp. 173–184, Apr. 1995.

[23] F. Tricas, F. García-Vallés, J. M. Colom, and J. Ezpeleta, "An iterative method for deadlock prevention in FMS," in *Discrete Event Systems*, vol. 569. Boston, MA, USA: Springer, 2000, pp. 139–148.

[24] S. A. Shah, E. L. J. Bohez, K. Shah, I. ul Haq, K. Azam, and S. Anwar, "Colored Petri net model for significant reduction of invariants in flexible manufacturing systems," *Int. J. Adv. Manuf. Technol.*, vol. 88, nos. 5–8, pp. 1775–1787, 2017.

[25] S. Liu, G. Liu, and H. Zhou, "A robust parallel object tracking method for illumination variations," *Mobile Netw. Appl.*, vol. 24, no. 1, pp. 1–17, 2018.

[26] C. Li, W. Wu, Y. Feng, and G. Rong, "Scheduling FMS problems with heuristic search function and transition-timed Petri nets," *J. Intell. Manuf.*, vol. 26, no. 5, pp. 933–944, 2015.

[27] F. Ahmad and S. A. Khan, "Module-based architecture for a periodic job-shop scheduling problem," *Comput. Math. Appl.*, vol. 64, no. 1, pp. 1–10, 2012.

[28] O. T. Baruwa and M. A. Piera, "Anytime heuristic search for scheduling flexible manufacturing systems: A timed colored Petri net approach," *Int. J. Adv. Manuf. Technol.*, vol. 75, nos. 1–4, pp. 123–137, 2014.

[29] J. P. Caballero-Villalobos, G. E. Mejía-Delgadillo, and R. G. García-Cáceres, "Scheduling of complex manufacturing systems with Petri nets and genetic algorithms: A case on plastic injection moulds," *Int. J. Adv. Manuf. Technol.*, vol. 69, nos. 9–12, pp. 2773–2786, 2013.

[30] L. Bianchi, M. Dorigo, L. M. Gambardella, and W. J. Gutjahr, "A survey on metaheuristics for stochastic combinatorial optimization," *Natural Comput.*, vol. 8, no. 2, pp. 239–287, 2009.

[31] S. Liu, Z. Pan, and X. Cheng, "A novel fast fractal image compression method based on distance clustering in high dimensional sphere surface," *Fractals*, vol. 25, no. 4, 2017, Art. no. 1740004.

[32] M. Gendreau and J.-Y. Potvin, "Metaheuristics in combinatorial optimization," *Ann. Oper. Res.*, vol. 140, no. 1, pp. 189–213, 2005.

[33] F. Pezzella, G. Morganti, and G. Ciaschetti, "A genetic algorithm for the flexible job-shop scheduling problem," *Comput. Oper. Res.*, vol. 35, no. 10, pp. 3202–3212, 2008.

[34] H. H. Xiong and M. C. Zhou, "Scheduling of semiconductor test facility via Petri nets and hybrid heuristic search," *IEEE Trans. Semicond. Manuf.*, vol. 11, no. 3, pp. 384–393, Aug. 1998.

[35] H. Yu, A. Reyes, S. Cang, and S. Lloyd, "Combined Petri net modelling and AI-based heuristic hybrid search for flexible manufacturing systems—Part II. Heuristic hybrid search," *Comput. Ind. Eng.*, vol. 44, no. 4, pp. 545–566, Apr. 2003.

[36] G. Mejia and N. G. Odrey, "An approach using Petri nets and improved heuristic search for manufacturing system scheduling," *J. Manuf. Syst.*, vol. 24, no. 2, pp. 79–92, 2005.

[37] B. Huang, R. Jiang, and G. Zhang, "Comments on 'Heuristic search for scheduling flexible manufacturing systems using lower bound reachability matrix,'" *Comput. Ind. Eng.*, vol. 67, no. 4, pp. 235–236, Nov. 2014.

[38] A. Reyes, H. Yu, G. Kelleher, and S. Lloyd, "Integrating Petri nets and hybrid heuristic search for the scheduling of FMS," *Comput. Ind.*, vol. 47, no. 1, pp. 123–138, 2002.

[39] S. Dauzère-Pérès and J. Paulli, "An integrated approach for modeling and solving the general multiprocessor job-shop scheduling problem using tabu search," *Ann. Oper. Res.*, vol. 70, pp. 281–306, Apr. 1997.

[40] M. Mastrolilli and L. M. Gambardella, "Effective neighbourhood functions for the flexible job shop problem," *J. Scheduling Banner*, vol. 3, no. 1, pp. 3–20, 2000.

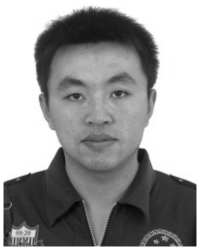
[41] J. Gao, L. Sun, and M. Gen, "A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems," *Comput. Oper. Res.*, vol. 35, no. 9, pp. 2892–2907, 2008.

[42] A. Ben Hmida, M. Haouari, M.-J. Huguet, and P. Lopez, "Discrepancy search for the flexible job shop scheduling problem," *Comput. Oper. Res.*, vol. 37, no. 12, pp. 2192–2201, 2010.

[43] X. Li and L. Gao, "An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem," *Int. J. Prod. Econ.*, vol. 174, pp. 93–110, Apr. 2016.

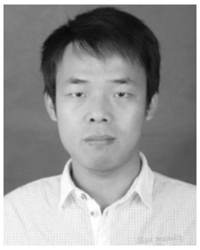
XIAOQIANG WU was born in Xingtai, Hebei, China, in 1985. He received the B.S. degree in electronic and information engineering from the Agricultural University of Hebei, Baoding, Hebei, China, in 2007, and the M.S. degree in agricultural electrification from Yunnan Agricultural University, Kunming, China, in 2013. He is currently pursuing the Ph.D. degree in mechanical engineering with the School of Mechanical Engineering, Tianjin University, Tianjin, China.

Since 2018, he has been an Associate Professor with the College of Mechanical Engineering, Inner Mongolia University for Nationalities. His research interests include advanced manufacturing technology, fundamental study of plasma sources, artificial intelligence, interdisciplinary application of computer, and the Internet of Things-enabled manufacturing systems.



SONGLING TIAN was born in Changzhi, Shanxi, China, in 1985. He received the B.S. degree in mechanical engineering from Shanxi Agricultural University, Jinzhong, Shanxi, China, in 2009, and the M.S. degree in mechanical engineering from Fuzhou University, Fuzhou, China, in 2013. He is currently pursuing the Ph.D. degree in mechanical engineering with the School of Mechanical Engineering, Tianjin University, Tianjin, China.

His research interests include the Internet of Things-enabled manufacturing systems, shop floor scheduling, cloud manufacturing, process planning, big data, artificial intelligence algorithm, innovative design methodology, and fluid machinery.



LEI ZHANG was born in Shijiazhuang, Hebei, China, in 1987. He received the B.S. degree in mechanical engineering from the Hebei Normal University of Science and Technology, Qinhuangdao, Hebei, China, in 2010, and the M.S. degree in mechanical engineering from Yanshan University, Qinhuangdao, in 2013, and the Ph.D. degree in mechanical engineering from Tianjin University, Tianjin, China, in 2018.

Since 2018, he has been a Lecturer with the School of Mechanical Engineering, Tianjin University of Commerce. His research interests include mechanical dynamics, robust control, and intelligent manufacturing.

• • •