Multiagent and Bargaining-Game-Based Real-Time Scheduling for Internet of Things-Enabled Flexible Job Shop

Jin Wang, Yingfeng Zhang, Yang Liu and Naiqi Wu

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2871346, IEEE Internet of Things Journal

Multi-agent and Bargaining-game-based Real-time Scheduling for Internet of Things-enabled Flexible Job Shop

Jin Wang, Yingfeng Zhang, *Member*, *IEEE*, Yang Liu, and Naiqi Wu, *Senior Member*, *IEEE*

***Abstract*—With the rapid advancement and widespread applications of information technology in the manufacturing shop floor, a huge amount of real-time data is generated, providing a good opportunity to effectively respond to unpredictable exceptions so that the productivity can be improved. Thus, how to schedule the manufacturing shop floor for achieving such a goal is very challenging. This work addresses this issue and a new multi-agent-based real-time scheduling (MARS) architecture is proposed for an Internet of Things (IoT)-enabled flexible job shop. Differing from traditional dynamic scheduling strategies, the proposed strategy optimally assigns tasks to machines according to their real-time status. A bargaining-game-based negotiation mechanism is developed to coordinate the agents so that the problem can be efficiently solved. To demonstrate the feasibility and effectiveness of the proposed architecture and scheduling method, a proof-of-concept prototype system is implemented with Java agent development framework (JADE) platform. A case study is used to test the performance and effectiveness of the proposed method. Through simulation and comparison, it is shown that the proposed method outperforms the traditional dynamic scheduling strategies in terms of makespan, critical machine workload, and total energy consumption.**

Index Terms—**Multi-agent, Internet of Things, Flexible job shop, Real-time scheduling**

Manuscript received March 16, 2018; revised June 16, 2018; accepted September 12, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 51675441, the Fundamental Research Funds for the Central Universities under Grant 3102017jc04001, the 111 Project Grant of NPU under Grant B13044, and the Science and Technology development fund (FDCT) of Macau under Grant 106/2016/A3. (Corresponding authors: Yingfeng Zhang and Niaqi Wu.)

J. Wang is with the School of Mechanical Engineering, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, china, and also with the School of Mechanical Engineering, Xi'an Aeronautical University, Xi'an, Shaanxi 710077, China (e-mail: wj19852004@qq.com).

Y. F. Zhang is with the School of Mechanical Engineering, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, China (e-mail: zhangyf@nwpu.edu.cn).

Y. Liu is with the Department of Management and Engineering, Linköping University, SE-581 83 Linköping, Sweden (email: yang.liu@liu.se).

N. Q. Wu is with the Institute of Systems Engineering, Macau University of Science and Technology, Macau, China, and also the national key laboratory of precise electronic manufacturing technology and equipment, Guangdong University of Technology, Guangzhou 510006, China (e-mail: nqwu@must.edu.mo).

I. Introduction

In today's highly competitive marketplace, manufacturing enterprises have to face enormous challenges such as the increased diversity in customer demands, globalized market, and environmental pressures [1][2]. With these challenges, to be competitive, a manufacturing system should have good flexibility, quick response, and fault-tolerant capability. Production scheduling plays an important role for a manufacturing system to improve productivity and responsiveness. A well-operated manufacturing system can increase capacity utilization and reduce lead time so as to increase profit gain [3][4]. Thus, in recent years, production scheduling has attracted great attention [5]-[17], especially for flexible job shop scheduling problems [18].

With a severely competitive market environment, traditional scheduling strategies with all tasks and manufacturing resources being controlled by a distribution system are no longer effective for an open, flexible, demand-driven, and reconfigurable manufacturing system [19]. The traditional scheduling strategies are intrinsically inflexible and not able to respond effectively to exceptional events (e.g., machine breakdowns and rush orders) and cannot adapt to unforeseen dynamic situations.

With the recent advancement of information technology, artificial intelligence (AI) has been developed prosperously. As an AI tool, multi-agent technology has been regarded as one of the most promising approaches for solving production scheduling problems and attracts great researchers' attention [20]. Unlike a traditional scheduling strategy driven by a centralized scheduler, a multi-agent-based scheduling system supports distributed scheduling, which is realized by autonomous agents. These agents collaborate and cooperate dynamically to optimize both local and global objectives [21]. Recently, investigations have been made by a number of scholars on multi-agent-based dynamic scheduling [22][23]. However, most of these researches mainly focus on the architectures of multi-agent systems (MAS) and negotiation protocols among the agents, as well as the application of distributed features of MAS for task allocation in a traditional manufacturing shop floor [24]. Few of them consider the real-time-data-based interaction between machines and other distributed resources in an IoT-enabled flexible job shop. As a result, often the performance of efficiency is degraded and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2871346, IEEE Internet of Things Journal

more energy is consumed by the production processes mainly due to the unpredictable exceptions [25].

Recently, the rapid progress of information technology (e.g. radio frequency identification-RFID) provides shop floor with rich real-time data for better operational management [26][27]. With these technologies being adopted, real-time-data-based traceability, visibility, and interoperability can be realized to improve the performance of shop floor planning, monitoring, and control. At present, by extending the IoT technologies such as RFID to manufacturing environment [28]-[30], real-time data have become more accessible and ubiquitous, contributing to a big data environment [31][32]. Thus, in a real-world manufacturing environment, the real-time data streams coming from IoT make it possible for one to discard the existing scheduling approaches and adopt the multi-agent-based dynamic scheduling techniques. More recently, great attention has been paid to the real-time-data-based optimization issue for shop floor in both academia and industry. For example, Zhang et al. [33] put forward a dynamic optimization method for shop floor material handling (DOM-SMH) based on real-time and multi-source manufacturing data. Zhang et al. [34] also proposed a dynamic optimization model for flexible job shop scheduling (DOM-FJSS) based on real-time data for cloud manufacturing (CMfg).

Although significant advancements have been achieved in using the real-time data for performance improvement, there are unsolved issues for how to apply real-time data-driven decision to MARS problem in a manufacturing big data environment due to the increasing process complexity, unpredictable exceptions, etc. These issues are summarized as follows.

(1) How to design a new and effective MARS architecture based on real-time data to implement real-time scheduling for an IoT-enabled flexible job shop. Recently, in many studies, multi-agent technology is adopted to deal with the dynamic scheduling problem 35]. However, how to integrate the real-time manufacturing information between the multi-agent-based dynamic scheduling system and the manufacturing execution system is still an open issue. This implies that, during the manufacturing execution stage, the real-time manufacturing information cannot be well captured such that manufacturing tasks are assigned to machines without considering their real-time status and processing capability. Thus, designing a new MARS architecture based on real-time data for the real-time-data-based scheduling system is critical and necessary for applications. Moreover, in designing an MARS architecture, it is better to use JADE as a platform, because of its advantages such as simplicity, code compactness, and graphical user interface.

(2) How to design a new multi-agent-based real-time task allocation strategy to implement real-time scheduling based on real-time data in an IoT-based manufacturing environment. In the existing multi-agent-based dynamic scheduling methods, a dynamic scheduling approach focuses on dynamic dispatching rules [36] and event-driven rescheduling policies [37]. Between them, event-driven rescheduling policies are used by most of the methods. By such methods, an action is triggered to respond to an exceptional event that changes the current system status. Then, the current schedule is revised to adapt to the new status caused by the exceptional events. By doing so, it may result in a new schedule that is totally different from the original one [38]. Thus, some operations that have not started yet under the previous schedule at the time of rescheduling may change their starting time sharply, which strongly affects the execution of other operations that are scheduled based on the original schedule and brings instability and undermines the process continuity [39]. Therefore, a new multi-agent-based real-time task allocation strategy should avoid or reduce the influence of the unpredictable exceptions based on the real-time data in an IoT-enabled flexible job shop.

(3) How to design a new negotiation mechanism for the MARS in an IoT-enabled flexible job shop. In general, there are many negotiation modes available. The most commonly used negotiation mechanisms are the contract net protocol (CNP) [40] and its modified versions [41]. However, both of these two protocols are communication intensive. A heavy communication load hinders the agents to respond to unpredictable exceptions in a dynamic scheduling system and makes agents spend more time for processing messages than focusing on decision making. This is especially true for a manufacturing shop floor in the internet of manufacturing things (IoMT) environment with the vast amount of data concurrency and exchange. Game theory-based negotiation mechanism can provide a useful framework for analyzing MAS. In both the bargaining game and the MAS, agents are considered to exhibit rational decision making, have asymmetric information, and work together to improve or maximize their utilities. Therefore, a bargaining-game-based negotiation mechanism is necessary to reduce the communication burden among the agents and improve the problem-solving efficiency.

To address the above-mentioned challenges, by taking the advantages of IoT and considering the requirements of real-time data-driven optimal decision making of a real-time scheduling system, a new MARS architecture is presented in this study to provide a new paradigm by extending the IoT to manufacturing field. Under this architecture, sensors can be embedded in the manufacturing resources such as operators, machines, pallets, materials, etc. Then, they can interact with each other during the execution stage. The exchanged information and their status can thus be tracked. Based on the real-time information from the resources, a multi-agent-based real-time task allocation strategy is proposed to timely eliminate the influences caused by exceptional events in the shop floor. The proposed method for the scheduling problem is computationally efficient, since by this method only one operation is selected for assigning to one machine at a time. In addition, compared with the traditional negotiation mechanism, the bargaining-game-based negotiation mechanism developed in this work can improve the interaction ability between agents and enhance the communication efficiency.

The rest of the study is organized as follows. Section II reviews the related literature. After the architecture of MARS is developed in Section III, Section IV presents each agent model.

The bargaining-game-based solution is given in Section V. In Section VI, a case study is used to verify the feasibility and applicability of the designed architecture of MARS, and an instance is tested to prove the effectiveness of the proposed method. Finally, conclusions and recommendations are summarized in Section VII.

II. Literature Review

As above mentioned, in this section, we briefly review the studies that are relevant to game-theory-based MARS problem in an IoT-enabled flexible job shop. They are classified into two categories: multi-agent-based scheduling and game theory for scheduling.

A. *Multi-agent-based scheduling*

Multi-agent technology has been reported to be very successful in a wide range of scheduling applications [42]. Shaw [43] pioneered the use of agents for flexible manufacturing system scheduling and factory control. Parunak [44] was another earliest one who developed a multi-agent-based manufacturing control system, which assigns an agent to each node in a control hierarchy. In recent years, MAS has been widely adopted in manufacturing applications because of its flexibility, reconfigurability, and scalability [45]. Multi-agent technology has also been considered to be one of the most promising approaches to the scheduling problem of complex and flexible manufacturing systems due to its distributed, autonomous, and dynamic nature.

Nowadays, more and more researchers and practitioners attempt to solve dynamic scheduling problems using the multi-agent technology. A recent survey on multi-agent-based scheduling was presented by Perez-Gonzalez and Framinan [46]. Savino et al. [47] studied the multiple-objective flow shop modeling and dynamic scheduling problem by using MAS in a production context that is characterized by diversified and high-volume production mix. Zhang and Wong [21] studied the flexible job shop scheduling/rescheduling problem under a dynamic environment with different types of disruptions. They developed a hybrid MAS negotiation mechanism and proposed an ant colony optimization approach. By these studies, many novel ideas are proposed for the applications of the multi-agent technology in dynamic scheduling. It is demonstrated that the agent technology is effective for solving complicated scheduling problems. Moreover, MAS has been successfully applied to dynamic flow shop scheduling [48][49], dynamic job shop scheduling [50][51], integrated planning and scheduling [52][53], dynamic flexible manufacturing systems [54], and automated guided vehicle (AGV) systems [55][56]. These studies show that multi-agent technology has been widely applied to resolve dynamic scheduling problems for traditional manufacturing shop floor.

With the development of science and technology, advanced technologies and management methods can be used to optimize the production processes and make a manufacturing shop floor intelligent. In recent years, RFID has been widely applied for supporting production and scheduling in manufacturing shop floor, where manufacturing resources with RFID facilities being attached are converted into smart manufacturing objects that are able to sense, interact, so that an IoMT environment is realized. With the vast amount of data that are produced and exchanged concurrently, the states of a manufacturing shop floor under the IoMT environment change dynamically in a real-time way [28]. The above-mentioned techniques in the existing studies for the traditional manufacturing shop floor are not able to adapt to such an IoMT environment. Thus, multi-agent-based dynamic scheduling should fully consider the real-time information exchange among the agents under the IoMT environment.

In addition, to the best of the authors' knowledge, research reports on multi-agent-based dynamic scheduling by using JADE are quite limited and many studies focus on the interaction of agents only and do not consider the implementation issues. By a rigorous literature search, it is found that only a handful of studies fall into this topic. Among them, the work done by Wang et al. [57] seems to be the most relevant one. They proposed a multi-agent-based approach with a filtered-beam-search-based heuristic algorithm being integrated to solve the dynamic scheduling problem in a flexible manufacturing system (FMS) shop floor based on JADE platform. Then, Chen and Chen [54] used multi-agent technology to construct a multi-section flexible manufacturing system model. Then, with dispatching rules being combined, the manufacturing environment is simulated based on the JADE framework. However, none of these studies considers the real-time manufacturing information of the shop floor.

B. *Game theory for scheduling*

The early game theory studies appeared in the economics literature introduced in the book "the theory of games and economic behavior" by Rowland [58]. Then, Nash extended the results and proposed the concept of Nash equilibrium (NE). In the few decades followed, many studies have been done, and most of them focus on the subject of medicine, economics, communication, and cloud manufacturing [59]-[61]. Currently, game theory is becoming more and more popular and has been gradually introduced to deal with production scheduling problems [62]. Game theory can be classified into cooperative and non-cooperative games. By using the cooperative game, Calleja et al. [63] studied the single machine job scheduling problem, where clients could have more than one job to be processed and a job could be of interest for different players using cooperative games. Han et al. [64] studied the flexible flow shop scheduling problem with component altering times (FFSP-CAT), which is a specific form of a flexible flow shop scheduling problem with sequence dependent setup time in a practical scenario. They constructed a repeated cooperative model and provided a theoretical analysis of a game. By using the non-cooperative game, Zhou et al. [65] constructed a game-theory-based mathematical model to schedule jobs in networked manufacturing environments, a new scheduling problem. Zhang et al. [34] put forward a dynamic optimization model for flexible job shop scheduling based on game theory and a new real-time scheduling strategy and method are proposed.

It can be seen from the above literature that there are many studies on scheduling problems from the viewpoint of game theory. However, only few of them use multi-agent technology. Diepl and Reaidy [66] investigated the means of co-ordination in a production system based on a hierarchical MAS using game theory. Reaidy et al. [67] proposed a negotiation methodology based on a MAS for heterarchical and complex manufacturing control systems. Agnetis et al. [68] addressed a deterministic scheduling problem, where two agents compete for the usage of a single machine. A significant shortcoming of these studies is that they describe only the coordination problem among the multiple agents from the viewpoint of software construction without quantitatively analyzing the interaction among the agents. Moreover at present, the existing work seldom focuses on the FJSS problem, especially in the real-time FJSS problem using bargaining game. Therefore, from the MARS point of view in a flexible job shop, the existing research is still at an infant stage and considerable progress has yet to come.

To address the above challenges, this study proposes a new MARS architecture to implement real-time data-driven optimization approach in an IoT-enabled flexible job shop based on the JADE framework using bargaining game. This study differs from the existing work in the literature in two folds: (1) a multi-agent-based real-time scheduling approach based on JADE platform is proposed, which takes the advantage of the real-time manufacturing information for an IoT-enabled flexible job shop; and (2) a bargaining-game-based coordination mechanism for MARS is developed by analyzing the interaction among the agents in a flexible job shop. The implementation of the proposed approach is expected to increase productivity, as well as flexibility and responsiveness for an IoT-enabled flexible job shop.

In addition, the authors' previous study has been conducted on the subject of game theory-based flexible job shop scheduling [25]. This study is different from the authors' previous one as follows.

(1) The authors' previous work used the dynamic game theory to deal with the conflict and competition among the multiple objectives in a multi-objective flexible job shop scheduling problem. In that study, a non-cooperative game is played only once and there is no binding contract, the payoff of each player in the Nash equilibrium solution may have less benefit than the other non-Nash equilibrium solution, resulting in non-collective rationality. With this observation, this study develops a bargaining-game-based coordination mechanism for the real-time scheduling in the flexible job shop to overcome the shortcoming of the previous work.

(2) Our previous work focused on multi-objective optimization method and was not for shop scheduling optimization from the viewpoint of a distributed system, while this study proposes a multi-agent-based real-time scheduling approach based on the JADE platform with the real-time manufacturing information being taken into consideration for an IoT-enabled flexible job shop.

III. Overview Of MARS Based On Real-time Data

This study mainly discusses the multi-agent-based real-time FJSS problem in a discrete manufacturing environment. The objective of the proposed MARS is to implement the interactive perception of distributed manufacturing resources by extending automatic identification (auto-ID) technologies and using multi-agent technology to process real-time scheduling and thus achieve real-time optimization of manufacturing tasks based on the real-time status of the machines.

A. The MARS strategy

In this study, a new MARS strategy is proposed. For better understanding, traditional scheduling strategies and the MARS strategy based on real-time data are described as follows, respectively.

By a traditional scheduling strategy, all the tasks are centrally assigned to the corresponding machines by a distribution system. The decision model is centralized, and machines do not interact with other distribution resources. As a result, the real-time state information of the distribution resources has not been considered. Hence, often a deviation between a plan and its execution is inevitable because of unpredictable exceptions. Moreover, the computational complexity is high as the number of tasks and machines increases.

With the MARS strategy based on the real-time data, by using multi-agent technology, each machine automatically sends its real-time state information to the system and requests tasks for processing. Tasks continually interact with machines. Then, tasks can be assigned to the most appropriate machines according to the real-time status of the machines. Since the task allocation is done in a real-time information-driven way and an allocation strategy is started only for the machines according to their real-time status. Thus, the deviation between a plan and its execution resulting from a traditional scheduling strategy can be largely eliminated via the MARS strategy.

B. The overall architecture of MARS

Based on the MARS strategy, an overall architecture of MARS for a flexible job shop is designed as seen in Fig. 1. It consists of two layers: the JADE middleware layer and the multi-agent layer. The JADE middleware layer provides a JADE runtime environment such that the agent registration, management, and interaction can be realized. Each running instance in the JADE runtime environment is called a container and it can contain several agents. The set of active containers is called a platform. There is a special container that is active all the time in the platform and it is called the main container. All other containers register into the main container as soon as they start. Once the platform is activated, the JADE default agents, including agent management system (AMS) and directory facilitator (DF) are instantiated. The AMS agent acts as a supervisor that controls the use of other agents to the platform; while the DF agent provides a default yellow page service in the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2871346, IEEE Internet of Things Journal

platform. In this study, there is a single main container only in the platform on which the agents are executed.

The multi-agent layer includes a number of application agents to fulfill the real-time scheduling based on real-time data of the shop floor. They are machine agent (MA), task agent (TA), task pool agent (TPA), real-time scheduling agent (RSA), and real-time monitor agent (RMA). Each type of agents can be implemented in the JADE platform. The main functions of these agents are described as follows.

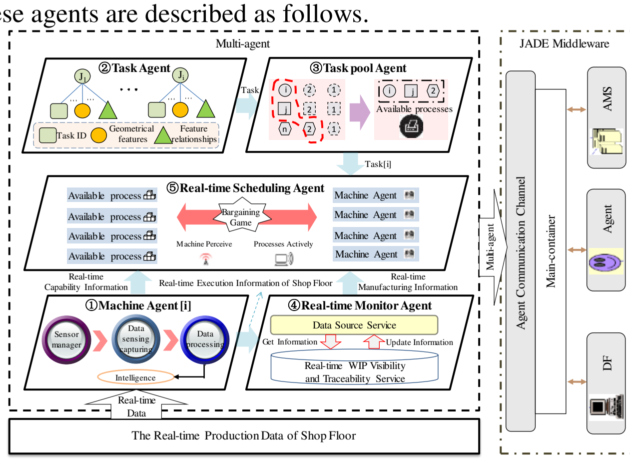


Fig.1. The Architecture of MARS

(1) The MA is responsible for capturing the real-time data sensed from auto-ID devices such as RFID and processing the complex real-time data such that they are understood as meaningful manufacturing information. Then, the corresponding agents can know the real-time status and available capacity of the manufacturing resources at any time.

(2) The TA is used to capture information of all tasks and send such information to TPA. If new tasks arrive, TA can capture this information timely and inform the TPA about the relevant conditions.

(3) The TPA is responsible for picking out the first unprocessed manufacturing operation of each task from the TA and publishing these available operations into the RSA timely.

(4) The RMA is responsible for capturing and processing the real-time production execution information of the shop floor and sending the real-time manufacturing information to the RSA. During the production execution, disturbances and changes of the shop floor processes are timely tracked and traced.

(5) The RSA provides a mathematic model and bargaining-game-based algorithm to optimally schedule the start time and finish time of each operation of each task according to the sensed real-time shop floor information.

The above five types of agents acquire related data by exchanging messages with each other. A message contains the following fields: the sender of the message, a list of receivers, the communicative act type, the message content expression, the content language, and the ontology. JADE provides the agent with communication language called Agent communication language message (ACLMessage). Messages exchanged by agents have a format specified ACL defined by the foundation for intelligent physical agents (FIPA) international standard for agent interoperability. A message in JADE can be implemented as an object of the jade.lang.acl.ACLMessage class that defines methods for handling all fields of a message.

C. *The implementation of MARS*

The implementation of MARS is the actual process of the interactions among the agents during real-time scheduling stage. Through auto-ID technologies, the real-time data can be captured by the MAs. Then, during the manufacturing execution stage, a task can be assigned to a most appropriate machine according to the machines' real-time status and available capability. The detailed process is described as follows.

When tasks are released to the shop floor for processing, the TA first captures all the specifications and processing conditions of each task. Further, this information is transferred to the TPA, which picks out the first unprocessed operation of each task and sends this information to the RSA. In this way, resources that are able to process these specified operations are known. At the same time, each MA automatically sends the real-time available capability information of corresponding machines to the RSA and the capable resources compete to process these operations. Consequently, the operations interact with the MAs continuously in the RSA. Thus, an operation can be assigned to the most suitable MA in an optimal way by using the bargaining game according to their real-time available capacity. Each time, only one operation is optimally assigned to the requested MA. MAs continuously send the request for new operations before all tasks are finished, which is released by the TPA. At the production execution stage, the RMA captures the real-time execution information of the shop floor and then sends real-time manufacturing information to the RSA. Therefore, if an exceptional event occurs, the manufacturing environment can be reconfigured. The RSA can decide the MAs that can continue to deal with the available operations or the tasks that should be removed or joined.

IV. MULTI-AGENT MODELS

A. *Machine Agent model*

Fig. 2 shows an MA model. It is responsible for wrapping the applications of manufacturing resources to capture the real-time data of manufacturing resources by adopting auto-ID and sensor technologies. Then, it processes the captured real-time data such that they can be understood as useful and meaningful manufacturing information. At each time *t*, the MAs actively send the real-time available capability information of a machine to the RSA for a machine to compete for processing the available operations according to their real-time status. An MA includes four modules, namely sensor manager, data sensing and capturing, data processing, and intelligent modules. The functions are described as follows.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2871346, IEEE Internet of Things Journal

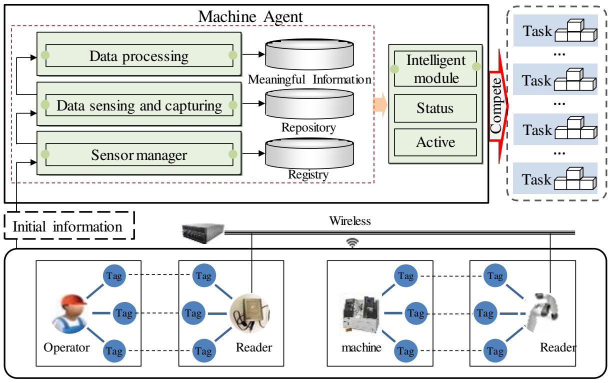


Fig.2. Machine Agent model

1) Sensor manager

This module is responsible for connecting and centrally managing the heterogeneous types of sensors for capturing the real-time data of manufacturing resources. First, it is used to register and manage (remote start, pause, stop, etc.) the behavior of sensors installed on a machine. Second, it is used to monitor and control the status of each registered sensor. Third, it is used to manage the capturing functions of each sensor and improve the sensing capability of each sensor. If a sensor breaks down, MAs can stop its behavior and send a message to the RMA.

2) *Data sensing and capturing*

This module is responsible for sensing and capturing the real-time data of the registered sensors installed on the manufacturing resources during the production process. Through the communication protocol and relationships in the registry, it can capture and transmit the sensed data from the sensors to the repository.

3) *Data processing*

This module is responsible for processing the insignificant data captured by registered sensors to form useful and meaningful information. Although real-time data record the real-time status of manufacturing resources, they need to be processed to provide useful and meaningful information. It can establish the mapping relation and mechanism such as rules and standard output data schemas to translate the real-time data to be meaningfully understood.

4) *Intelligent module*

Based on the useful and meaningful information, each MA can actively send the real-time available capability information and real-time status to the RSA and decide whether to compete for processing the tasks from the TPA.

B. *Task Agent model*

The TA is responsible for capturing the real-time information of each task and sending the information to the TPA. The TA model is shown in the top of Fig. 3. When tasks are released to the shop floor for processing, specifications and processing conditions of each task are registered by the TA. These specifications include task ID, the materials for the task, the hardness of the material, a list of geometrical features, feature relationships, estimated removal volume, tolerance, chip breakability, and surface quality requirements and so on. The processing conditions include different processing time and cutting power of each operation on different machines. Then, this real-time task information is transferred to the TPA.

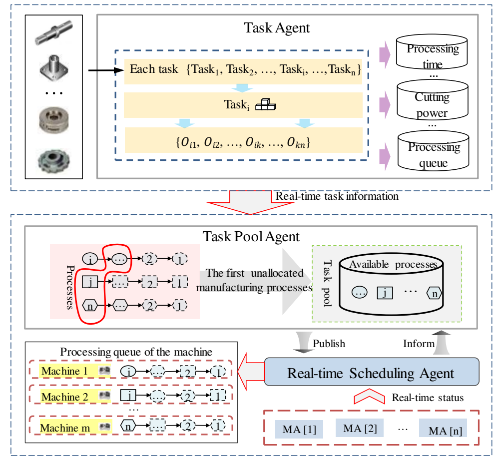


Fig.3. Task Agent and Task Pool Agent model

C. *Task Pool Agent model*

Based on the real-time task information from the TA, the TPA is used to pick out the available operations and publish these operations to the RSA at each time *t*. As seen in the lower part of Fig. 3, the work logic of the TPA includes three stages.

At the beginning, the TPA establishes a task pool and puts the first unallocated manufacturing operation of each task into it after receiving the real-time task information from the TA. Then, these available operations in the task pool are published into the RSA and each MA automatically sends its real-time status and requests to undertake the available operations from the RSA. If the previous operation in the RSA is submitted to the processing queue of a machine, the RSA informs the TPA and a new operation that belongs to the next manufacturing step is added into the task pool and then published into the RSA again. This process is repeated until all operations are added to the processing queue of appropriate machines.

D. *Real-time Monitor Agent model*

The RMA plays a key role for capturing the real-time execution information and sending it to the RSA. Fig. 4 shows the work logic of an RMA. There are mainly two modules in the RMA.

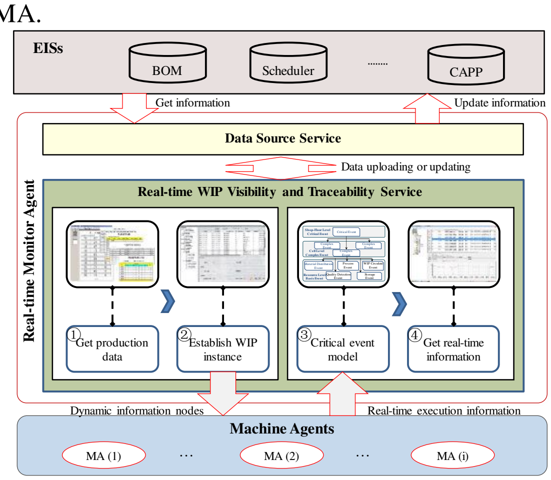


Fig.4. Real-time Monitor Agent model

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2871346, IEEE Internet of Things Journal

1) *Real-time WIP visibility and traceability service (RTWIP-VTS)*

RTWIP-VTS acts as a core in the RMA. On the one hand, it is responsible for establishing WIP instance by extracting the necessary information or updating the changed information from or to heterogeneous EISs (Enterprise Information Systems) through data source service. On the other hand, it is also responsible for modeling the dynamic behavior of a manufacturing system and processing the real-time execution information from a large amount of the low-level events captured by MAs. Here, the critical event model can extract the key information from the above low-level events to form high-level events. Through these high-level events, the information from the corresponding equipment such as dynamical status and produced WIP items can be monitored.

The inputs include BOM, schedule, process plans, and real-time execution information captured by relevant MAs. The outputs are the real-time manufacturing information related to produced products, consumed materials, exceptions, etc. of individual manufacturing resources, and the overall real-time production progress and production disturbances, etc. of the entire shop floor.

2) *Data source service*

The objective of data source service is to build a bridge for communication between the RMA and heterogeneous EISs. It provides data uploading, downloading, query, processing, and updating functions for sharing and integrating data between the RTWIP-VTS and other services or EISs. Due to the difficulties of information exchanging among the heterogeneous EISs, XML data with industrial standards are adopted for standardization to provide standardized schemas for manufacturing elements. The inputs of this module are the parameters of the data source of the EISs, while the outputs are the standard information based on XML schemas.

E. Real-time Scheduling Agent model

The RSA is designed to implement the real-time scheduling. At each time t, its inputs include the real-time capability information from MAs, available operations from the TPA, and real-time manufacturing information from the RMA. Its outputs are the task queues of the machines. Two modules, namely problem formulation module and solving module are involved in the RSA.

1) *Problem formulation module*

The FJSS problem can be formulated as follows. There is a set of n tasks to be processed on a set of m machines. Task i consists of a sequence of n_i operations. Each operation O_{ij} of task i can be processed by some capable machines. The FJSS problem is to optimally assign the operations to machines and sequence the operations assigned to each machine such that the given criteria are satisfied.

Based on the notation listed in Table I, a mathematic formulation for the problem is built, which is described as follows.

TABLE I
NOTATIONS

Notations	Description
n	the total number of tasks
m	the total number of machines
n_i	the total number of operations of task i
$M=\{M_1,M_2,\ldots,M_m\}$	the set of machines
O_{ij}	the j^{th} operation of task i
C_{ij}	the completion time of O_{ij}
C_M	the maximal completion time of the machines
W_k	the workload of M_k
W_M	the critical machine workload, which is the machine with the heaviest workload
E	the total energy consumption of production
x_{ijk}	1, if M_k is selected for O_{ij}; 0, otherwise
P_{0k}	the idle power of M_k (kW)
P_k	the cutting power of M_k (kW)
t_{Ik}	the total idle time of M_k
t_{ijk}	the processing time of O_{ij} on M_k

Objective function:

$$\text{Min} \quad f_1 = C_M = \text{Max } C_{ij} \quad i \in [1,n],\ j \in [1,n_i] \tag{1}$$

$$\text{Min} \quad f_2 = W_M = \text{Max}\{W_k\} = \text{Max}\{\sum_{i=1}^{n}\sum_{j=1}^{n_i}(t_{ijk}x_{ijk})\} \tag{2}$$

$$\text{Min} \quad f_3 = E = \sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n_i}(P_k t_{ijk} x_{ijk}) + \sum_{k=1}^{m}(P_{0k} t_{Ik}) \tag{3}$$

Subject to:

$$\begin{aligned} & C_{i,j} - C_{i,j-1} \ge t_{i,j,k} \cdot x_{i,j,k} \qquad j = 2,\cdots,n_i \\ & C_{i,j} > 0 \quad i = 1,2,\cdots,n \end{aligned} \tag{4}$$

$$\sum_{k \in M(O_{ij})} x_{ijk} = 1 \qquad \forall i, j \tag{5}$$

For Objectives (1)–(3), f_1 represents makespan or the maximal completion time of the machines, f_2 represents the critical machine workload, and f_3 represents the total energy consumption for producing the tasks. These objectives are changed as scheduling result changes. Hence, by minimizing these objectives, an optimal schedule can be obtained at each time t. Inequality (4) ensures the operation precedence constraints. Constraint (5) guarantees that an operation is assigned to one and only one machine.

2) *Bargaining-game-based solving module*

It follows from the above formulation that the multi-agent-based real-time scheduling problem is a multi-objective optimization problem (MOP). A general MOP can be summed up in the following common mode:

$$\begin{cases} \min/\max \ f_1(x) \\ \quad \cdots \quad \cdots \quad \cdots \\ \min/\max \ f_i(x) \\ \quad \cdots \quad \cdots \quad \cdots \\ \min/\max \ f_k(x) \\ s.t. \ g_j(x) \le 0, \ j = 1,\cdots,m_1 \\ \qquad h_l(x) = 0, \ l = 1,\cdots,m_2 \end{cases} \tag{6}$$

where $x = (x_1,\cdots,x_n) \in X$ is a decision variable, X is the variables space, $f_i(x), (i = 1,2,\cdots,k)$ is a cost function, $g_j(x)$ and $h_l(x)$ together refer to as the constraints.

For the MOP (Equation (6)), $f_i(x), (i = 1,2,\cdots,k)$ can be regarded as the k players in a bargaining game. The decision strategy space S equals to a variable space X. The payoff function for each player is $f_i(x)$. The bargaining game equilibrium solution can be seen as a solution for the MOP. Thus, this module is used to calculate the optimal solution by adopting the bargaining game. The bargaining-game-based solution includes players, strategies, and payoff design, and a bargaining game equilibrium solution. The details of the bargaining-game-based solution are described next.

V. Bargaining-Game-Based Solution in RSA

A. *Bargaining game model*

The multi-agent-based real-time scheduling problem addressed in this study can be seen as an N-person bargaining game with complete information. Bargaining game is defined as that decision-makers solve the profit distribution problem through consultation. To build a bargaining game model, three elements should be determined: players, strategies, and payoff, which can be described as:

$$G = \{F_i; S_i; U_i\} \quad i = 1,2,3 \tag{7}$$

where F_i is the set of players who participate in the bargaining game. In the problem addressed in this work, the three objectives correspond to three players. Here, players take actions sequentially, and the choice made by the former player has an impact on the selection made by the latter.

S_i is the actions or strategies adopted by Player i. In this problem, the available operations from the RSA to the strategies of this game are denoted as strategy profile, meaning that the first unprocessed manufacturing operations of the tasks are strategies for players at each time t.

U_i is the payoff function for Player i. In the addressed problem, the utility functions for the three players are the first, second, and third objective functions, respectively.

B. *The bargaining-game-based real-time scheduling method*

At each time t for a real-time schedule, a bargaining-game-based real-time scheduling method is triggered in the RSA such that the operations can be assigned to the most suitable MA according to the real-time available capability information of the MAs. At each time t, the problem-solving procedure is described as follows.

Step 1: MAs are assigned to the three objectives in turn. For example, MA[1] is for f_1 and MA[2] for f_2 and so on. In a real-world manufacturing system, the number of MAs is greater than three, so we can assign MA[4] to f_1 and MA[5] to f_2 until all MAs are assigned to an objective.

Step 2: Three objectives correspond to three players. Each player tries to select the most appropriate operations such that the goal of maximizing its payoff is achieved according to the results of the negotiation. Here, there are many stages in the bargaining game, and each stage has one player or one MA[i] to make a decision. Therefore, each MA that is assigned to f_i can choose an available operation from the RSA.

Step 3: Calculate the utility functions $u_1(s)$, $u_2(s)$ and $u_3(s)$ for Players 1, 2, and 3 according to Eqs. (1) - (3) from each feasible strategy combination, respectively.

Step 4: Find the bargaining game equilibrium solution, which is described in detail in Part C of Section V. Then, the available operations in the RSA are assigned to the most suitable MAs in an optimal way according to their real-time status.

Step 5: At the next time t (t=t+1), repeat Steps 1 - 4 until all the tasks are assigned.

When exceptional events (e.g., machine breakdown, change of the order, etc.) occur in a real-time, the influences of the exceptions can be timely reduced and eliminated through changing the players or the strategies of the bargaining game.

C. *Bargaining game equilibrium solution*

Sub-game perfect Nash equilibrium (SPNE) is broadly considered and applied as the solution for N-person non-cooperative dynamic game. An SPNE point is an N-tuple of strategies, one for each player, such that anyone who deviates from it unilaterally cannot possibly improve its expected payoff. Compared with the dynamic game, bargaining game is a process of value creating and redistributing, and the final agreement allows players to get a higher payoff than bargaining before. Watson [69] has presented the standard solution for bargaining problem. However, he focused on a two-player case only. In this section, an algorithm based on the solution of Watson is put forward to search for the bargaining game equilibrium solution with three players.

Let V denote the set of payoff vectors defining the players' alternatives for the bargaining game.

$$V = \{u_i(s^1), u_i(s^2), \cdots, u_i(s^k), \cdots, u_i(s^n)\} \; i = 1,2,3 \tag{8}$$

$$s^k = \{s_1^k, s_2^k, s_3^k\}$$

Let d denote the payoff vector associated with the default outcome, which describes what happens if the players fail to reach an agreement, $d \in V$. In this paper, d is given by $u_i(s^*)$ and $s^* = (s_1^*, s_2^*, s_3^*)$ is one SPNE solution for the bargaining game.

Let $u^*(s^k)$ denote the maximized joint value for the bargaining game.

$$u^*(s^k) = \max \sum_{i=1}^{3} u_i(s^k) \tag{9}$$

There are cases where the default payoff is the largest one, i.e., $u^*(s^k)$ is the default payoff.

Let p denote the surplus of an agreement, which is defined as the difference between the joint value of the contract and the one obtained when the players do not reach an agreement. We have

$$p = u^*(s^k) - \sum_{i=1}^{3} u_i(s^*) \tag{10}$$

Let π_i be the proportion of p obtained by Player i. When an agreement is reached such that each player obtains the final payoff as:

$$u_i^{final}(s^k) = u_i(s^*) + \pi_i p \tag{11}$$

$$\pi_i \geq 0$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2871346, IEEE Internet of Things Journal

$$\sum_{i=1}^{3}\pi_i = 1$$

Let t denote the contracted monetary transfer.

$$t = | u_i^{final}(s^k) - u_i^*(s^k) | \qquad (12)$$

Thus, a bargaining solution can be computed by the algorithm briefly summarized in Fig. 5.

By this algorithm, the bargaining game equilibrium solution s^k is found. For Player i, the payoff is $u_i^{final}(s^k)$. The contracted monetary transfer between the players is t. In addition, this algorithm can be extended to find bargaining game equilibrium solutions for the N-person bargaining game.

// Algorithm for bargaining game equilibrium solution
Input: A bargaining game procedure.
Start
Step 1. Calculate the maximized value $u^*(s^k)$ by determining the value s^k that maximizes $u_1(s^k)+u_2(s^k)+u_3(s^k)$;
Step 2. Determine an SPNE according to each player's utility using backward induction. The contents of the SPNE solution are described in the authors' previous paper [25].
Step 3. Player i obtains the payoff
$$u_i^{final}(s^k) = u_i(s^*) + \pi_i(u^*(s^k) - u_1(s^*) - u_2(s^*) - u_3(s^*))$$
where $u_i(s^*)$ is player i's default payoff.
Step 4: Calculate t to find the transfer that achieves the required split of the surplus.
END
Outputs: s^k and $u_i^{final}(s^k)$.

Fig.5. Solution procedure

VI. Case Study

To demonstrate the applicability and efficiency of the proposed approach for MARS, a proof-of-concept prototype system is built on the JADE platform with the Netbeans 8.0 development environment. The bargaining-game-based real-time scheduling method is coded in Java and is encapsulated into the RSA. Experimental simulations are conducted in the prototype system on Intel Core i5 3.10 GHz PC with 8GB RAM memory. Simulation results with comparisons are also given.

A. *Case Scenario*

The case scenario is about an FMS. For simplicity of understanding and without loss of generality of principle, basic manufacturing resources are selected for configuring a practical proof-of-concept demonstration. As shown in the lower part of Fig. 6, this demo manufacturing environment consists of the following main components, namely a raw material area and a finished product area for storing materials, WIP and finished products; a manufacturing area with eight machines and tasks to be processed. Each machine is capable of active sensing, interaction, and self-decision. In order to acquire real-time data during the manufacturing execution stage, tags are attached to some types of manufacturing resources.

The JADE-platform-based MAS is constructed as seen in the upper part of Fig. 6. An MA can capture the real-time data of the shop floor by equipping auto-ID devices. Then, the real-time capability and execution information of the manufacturing resources can be timely sensed by the RSA and RMA, respectively. The TA receives tasks from EISs as soon as production orders are released into the shop floor. It records the information of tasks and sends to the TPA. The TPA picks out the available operations of each task and publishes these available operations into the RSA. Then, the RSA optimally schedules the start time and finish time of each operation of each task according to the sensed real-time shop floor information. The RMA gets the necessary manufacturing information relevant to the production orders from EISs and real-time execution information from the MA.

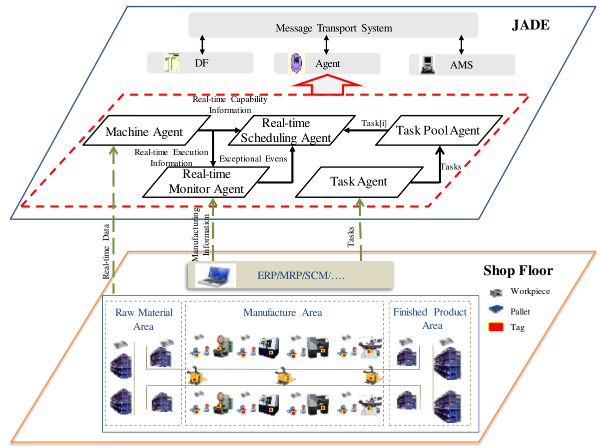


Fig.6. Diagram of the case scenario

To simulate this case scenario, a simple experimental system is established according to the case scenario as shown in Fig.7. The experimental system is composed of two Industrial Personal Computers (IPCs), a number of RFID readers, and a mass of tags. These readers are connected to the IPCs and each reader connects six antennas. The antennas are placed in the corresponding locations for capturing real-time data from different manufacturing resources. These tags are grouped into five types, namely equipment, operators, pallets, critical tools, and WIP items to simulate the real-time events of machines, automatic guided vehicles (AGVs), industrial robots (IRs), operators, pallets, tools, materials, WIP items, and finished products. The real-time status of machines, AGVs, IRs, operators, pallets, tools, materials, WIP items, and finished products can be easily captured from their tags or a special strategy.

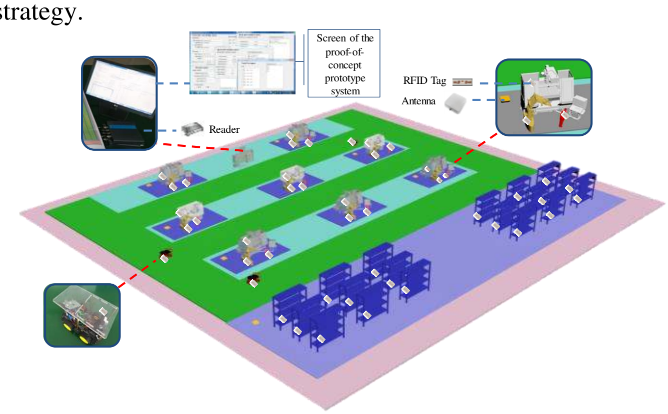


Fig.7.Simulation experiment of the case scenario

B. *Experimental trials*

Based on the above-mentioned prototype system, this section illustrates the MARS through a simulation example.

The scheduling problem is based on Kacem's instance [70], where there are eight machines for processing eight tasks with totally 27 operations. Compared with Kacem's instance, to optimize the total energy consumption for completing these tasks, we present the cutting power required for the operations on different machines. The detailed information about the tasks is shown in Table II. In Table II, (x/y) in Row O_{ij} and Column M_k mean the time taken for processing operation 'j' of task 'i' by machine 'k' is 'x' and its cutting power is 'y'. For example, (3/1.8) in row O_{11} and column M_2 means that the time taken for processing operation '1' of task '1' on machine '2' is '3', and its cutting power required is '1.8'. Table III gives the power required when a machine is idle, which is abstracted from research work developed by [71]. The time unit is defined as hours, and the cutting power unit is defined as kWs.

TABLE II
THE INSTANCE OF MARS

Tasks	Processes	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
	O_{11}	5/1.3	3/1.8	5/3.2	3/1.1	3/1.1	-	10/0.8	9/1.1
J_1	O_{12}	10/1.3	-	5/3.4	8/3.2	3/0.8	9/0.8	9/0.9	6/1.3
	O_{13}	-	10/1.8	-	5/1.4	6/0.7	2/0.9	4/1.2	5/1.3
	O_{21}	5/1.6	7/2.1	3/2.6	9/1.5	8/1.2	-	9/1.1	-
J_2	O_{22}	-	8/2.4	5/2.4	2/1.6	6/1.4	7/1.2	10/1.3	9/1.4
	O_{23}	-	10/2.3	-	5/1.5	6/0.9	4/1.8	1/1.4	7/1.3
	O_{24}	10/1.4	8/1.8	9/2.4	6/3.2	5/0.8	7/1.7	-	-
	O_{31}	10/2.1	-	-	7/1.5	6/0.7	5/1.6	2/1.3	4/1.2
J_3	O_{32}	-	10/1.9	6/2.6	4/1.6	8/1.2	9/1.7	10/1.4	-
	O_{33}	1/1.4	4/2.5	5/4.2	6/1.4	-	10/1.3	-	7/0.8
	O_{41}	3/1.3	1/2.4	6/3.2	5/2.1	9/1.3	7/1.7	8/1.3	4/1.1
J_4	O_{42}	12/1.4	11/2.6	7/4.2	8/3.2	10/1.5	5/0.8	6/1.2	9/1.3
	O_{43}	4/1.4	6/3.7	2/3.2	10/1.5	3/0.8	9/0.7	5/1.4	7/1.8
	O_{51}	3/1.3	6/1.2	7/2.4	8/1.2	9/0.8	-	10/1.3	-
J_5	O_{52}	10/1.2	-	7/2.8	4/2.1	9/1.3	8/0.7	6/1.3	-
	O_{53}	-	9/3.2	8/3.2	7/1.8	4/1.2	2/1.2	7/1.4	-
	O_{54}	11/2.1	8/1.6	-	6/1.7	7/1.5	5/1.3	3/1.3	6/1.3
	O_{61}	6/1.4	7/1.7	1/4.2	4/1.6	6/0.8	9/1.4	-	10/1.3
J_6	O_{62}	11/1.3	-	9/3.2	9/1.4	9/0.9	7/0.9	6/1.3	1/1.3
	O_{63}	10/1.4	5/2.1	9/2.4	10/1.5	11/1.2	-	10/1.2	-
	O_{71}	5/1.1	4/2.2	2/3.2	6/1.3	7/1.3	-	10/0.8	-
J_7	O_{72}	-	9/2.5	-	9/1.4	11/0.8	9/1.6	10/1.3	5/1.4
	O_{73}	-	8/2.4	9/4.2	4/1.2	8/1.2	6/2.1	-	10/1.6
	O_{81}	2/1.4	8/3.2	5/2.2	9/1.4	-	4/1.2	-	10/1.8
J_8	O_{82}	7/1.3	4/1.7	7/2.9	8/1.4	9/1.1	-	10/1.3	-
	O_{83}	9/1.4	9/3.2	-	8/1.2	5/0.8	6/1.3	7/1.4	1/1.3
	O_{84}	9/1.7	-	3/4.1	7/1.2	1/0.9	5/1.4	8/0.9	-
	$O_{9,1}$	5/1.3	7/2.4	8/3.2	5/1.1	5/1.1	-	7/1.6	4/1.2
J_9	$O_{9,2}$	4/1.6	7/2.3	14/2	4/1.2	3/1.3	6/1.1	-	10/1.3
	$O_{9,3}$	5/1.2	4/2.2	6/3.2	11/1.3	7/2.1	13/1.3	5/1.3	5/1.5
	$O_{10,1}$	2/1.6	-	4/3.5	-	7/2.2	5/3.2	4/1.8	6/3.5
J_{10}	$O_{10,2}$	8/4.3	-	8/2.1	1/3.6	-	5/4.3	8/2.9	7/3.4
	$O_{10,3}$	7/7.1	3/2.2	-	4/1.5	8/4.1	1/1.5	8/2.2	4/0.8

TABLE III [71]
IDLE POWER OF MACHINES

M_k	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
Idle power [kW]	0.995	1.485	1.91	0.6	0.43	0.56	0.47	0.72

In this case, based on real-time data, the execution procedure for MARS includes mainly five steps as shown in Fig. 8 and it is described in detail as follows.

At the beginning, when an operator comes to a machine and starts it up, the information relevant to this machine is sensed. The MAs capture the real-time manufacturing data. Then, each MA automatically sends its real-time available capability information to the RSA and real-time execution information to the RMA.

When the tasks are released into the system, processing details for all operations are captured by the TA and are put into the TPA. Then, the TPA picks up the first unprocessed manufacturing operation of each task and put them into the RSA.

Consequently, the RSA knows the real-time capability information from MAs and the available operation information from the TPA. Thus, the available operations interact with MA[i] continuously and can be assigned to the most suitable machines in an optimal way using bargaining game as described in Section V according to their real-time status. Here, $\pi_i = \frac{1}{3}$.

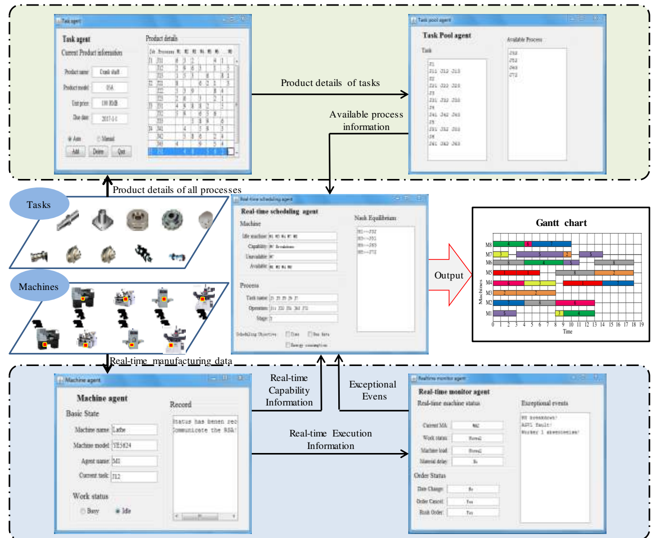


Fig.8. The procedure of real-time production scheduling based on the multi-agent technology

These steps are repeated until all the operations are assigned to certain machines for processing. Table IV shows the procedure of real-time scheduling at each time t.

TABLE IV
THE PROCEDURE OF REAL-TIME SCHEDULING WITHOUT CONSIDERING THE EXCEPTIONAL EVENTS

Time	Idle machines	Optional process(es)	Real-time scheduling result(s)
0	$\{M_1,M_2,M_3,M_4,M_5,M_6,M_7,M_8\}$	$\{O_{11}, O_{21}, O_{31}, O_{41}, O_{51}, O_{61},O_{71},O_{81}\}$	$O_{51}\rightarrow M_1$; $O_{71}\rightarrow M_2$; $O_{21}\rightarrow M_3$; $O_{61}\rightarrow M_4$; $O_{11}\rightarrow M_5$; $O_{81}\rightarrow M_6$; $O_{31}\rightarrow M_7$; $O_{41}\rightarrow M_8$
1	None	None	None
2	$\{M_7\}$	$\{O_{32}\}$	None
3	$\{M_1,M_3,M_5,M_7\}$	$\{O_{12}, O_{22}, O_{32}, O_{52}\}$	$O_{22}\rightarrow M_3$; $O_{12}\rightarrow M_5$; $O_{52}\rightarrow M_7$
4	$\{M_1,M_2,M_4,M_6,M_8\}$	$\{O_{32}, O_{42},O_{62}, O_{72}, O_{82}\}$	$O_{82}\rightarrow M_2$; $O_{32}\rightarrow M_4$; $O_{42}\rightarrow M_6$; $O_{62}\rightarrow M_8$
5	$\{M_1, M_8\}$	$\{O_{63}, O_{72}\}$	$O_{72}\rightarrow M_8$
6	$\{M_1,M_5\}$	$\{O_{13}, O_{63}\}$	None
7	$\{M_1,M_5\}$	$\{O_{13}, O_{63}\}$	None
8	$\{M_1,M_2,M_3,M_4,M_5\}$	$\{O_{13}, O_{23}, O_{33}, O_{63}, O_{83}\}$	$O_{33}\rightarrow M_1$; $O_{63}\rightarrow M_2$; $O_{83}\rightarrow M_5$
9	$\{M_1,M_3,M_4,M_6,M_7\}$	$\{O_{13}, O_{23}, O_{43}, O_{53}\}$	$O_{43}\rightarrow M_1$; $O_{13}\rightarrow M_4$; $O_{53}\rightarrow M_6$; $O_{23}\rightarrow M_7$
10	$\{M_3,M_7,M_8\}$	$\{O_{24}, O_{73}\}$	None
11	$\{M_3, M_6,M_7,M_8\}$	$\{O_{24}, O_{54}, O_{73}\}$	$O_{54}\rightarrow M_7$
12	$\{M_3, M_6, M_8\}$	$\{O_{24},O_{73}\}$	None
13	$\{M_1,M_2,M_3,M_5,M_6, M_8\}$	$\{O_{24},O_{73},O_{84}\}$	$O_{24}\rightarrow M_5$; $O_{84}\rightarrow M_6$
14	$\{M_1,M_2,M_3,M_4, M_7,M_8\}$	$\{O_{73}\}$	$O_{73}\rightarrow M_4$

During the production execution stage, the real-time execution information of the shop floor is captured by the RMA and sent to the RSA. If exceptions occur, the RSA can be certainly noticed the change caused by such exceptions according to the dynamic manufacturing environment. Thus, the RSA can respond to them timely such that the influence brought by the exceptions can be greatly reduced or even eliminated.

To validate the effectiveness in responding to the exceptional events under the real-time shop floor environment by the proposed MARS method, we compare it with several traditional dynamic scheduling methods, including complete reactive scheduling method. By the complete reactive scheduling method, operations are assigned to machines according to a specific assignment rule. Then, once a machine becomes available and there are operations in its waiting queue, it chooses the operations with the highest priority to process based on a heuristic priority dispatching rule.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2871346, IEEE Internet of Things Journal

Two popular priority dispatching rules are employed. They are shortest processing time (SPT) and first-in-first-out (FIFO). Also, we consider two machine assignment rules for comparison. The first one finds the available machine with the minimum processing time for an operation and then that the operation is assigned to this machine. The second one assigns an operation to its alternative machine which has the minimum workload currently. We call them MAR1 and MAR2 in short.

To make comparisons, the simulation results for the three test cases are summarized in Tables V- VII. For Test Case 1, two exceptions happen during the production execution stage, i.e., M_1 and M_6 are broken down at time t_1=4 and t_2=6, they are repaired at time t_3=6 and t_4=8, respectively. For Test Case 2, rush Tasks 9 and 10 are added at time t_5=2 and t_6 =4, respectively. For Test Case 3, four exceptions occur, i.e., M_1 and M_6 are broken down at time t_7=4 and t_8=6, and they are repaired at time t_9=6, t_{10}=8, respectively; furthermore, two rush Tasks 9 and 10 are added at time t_{11} =2 and t_{12} =4, respectively.

TABLE V
TEST CASE 1

Scheduling methods	C_M [hour]	W_M [hour]	E [kW·h]
MAR1+SPT	24	22	242.02
MAR1+ FIFO	24	22	216.97
MAR2+ SPT	25	23	249.26
MAR2+ FIFO	23	22	226.85
Proposed method	18	16	156.12

TABLE VI
TEST CASE 2

Scheduling methods	C_M [hour]	W_M [hour]	E [kW·h]
MAR1+SPT	39	34	364.72
MAR1+ FIFO	30	27	387.62
MAR2+ SPT	35	31	410.68
MAR2+ FIFO	32	27	374.49
Proposed method	20	20	191.44

TABLE VII
TEST CASE 3

Scheduling methods	C_M [hour]	W_M [hour]	E [kW·h]
MAR1+SPT	25	23	299.50
MAR1+ FIFO	27	27	301.26
MAR2+ SPT	32	29	368.92
MAR2+ FIFO	31	31	328.42
Proposed method	25	23	217.92

The simulation results for Case 1 are given in Table V. The scheduling results obtained by our methods have better solutions compared to the traditional dynamic scheduling method. For the solutions obtained by the proposed method, C_M is 18 hours, while it is 23 and 25 hours for the best and worst ones obtained by the traditional dynamic scheduling method. The maximum improvement is 28.0% and the minimum improvement is 21.7%. The minimum and maximum values of W_M obtained by the traditional dynamic scheduling method are 22 hours and 23 hours, respectively. Thus, the proposed method improves W_M by 27.3% and 30.4%, respectively, for its minimum and maximum values than the traditional one. Compared with the traditional dynamic scheduling method, the proposed method reduces E for the maximum value by 249.26 kW·h and the minimum value by 216.97 kW·h, i.e., reduces it by 37.4% and 28.0%, respectively.

The simulation results from test Case 2 are given in Table VI. It can be seen that, for C_M, it is 20 hours by the proposed method, while, by the traditional dynamic scheduling method, it is 30 hours and 39 hours for the best and worst values. Thus, by the proposed method, it is improved by 33.3% and 48.7%, respectively. For W_M, by the proposed method, it is 20 hours, which means that, compared with the traditional dynamic scheduling method, the minimum improvement is 11.1% and the maximum improvement is 41.2%, respectively. In addition, for E, by the proposed method, it is 191.44 kW·h, which means that compared with the traditional dynamic scheduling method, a 47.6-53.4% improvement in the total energy consumption of production is achieved.

Test Case 3 can be seen as a variation of test Cases 1 and 2, where certain machines are broken down and rush orders are added at the same time. The simulation results are given in Table VII. It can be observed that the values of C_M and W_M obtained by the proposed method are the best values obtained from the traditional dynamic scheduling method. However, compared with the worst ones obtained by the traditional dynamic scheduling method, by the proposed method, it is improved by 21.9% and 25.8%, respectively. In terms of the total energy consumption, the proposed method also achieves better performance than the traditional dynamic scheduling method.

Thus, it follows from the above simulation results that, by the advanced IoT technology and optimization method, the critical performance indices for the MARS problem can be significantly improved. Also, the proposed method contributes to the sustainable development of manufacturing industry, especially in MARS.

VII. CONCLUSIONS

Recently, auto-ID technology has been widely adopted in the manufacturing shop floor. Such an automatic data collection approach brings new opportunities for better operations of shop floor at the one hand. However, it presents new challenges at the other hand. For example, how to develop a real-time-data-based real-time scheduling system for improving the performance of shop floor planning, execution, and control is a new issue and there is no applicable method. In this study, to address this issue, an architecture of MARS for a flexible job shop is presented to provide a new paradigm for manufacturing enterprises to enhance the efficiency of real-time scheduling so that the influence of exceptional events can be reduced. Based on this architecture, a bargaining-game-based real-time scheduling strategy is proposed to implement real-time scheduling. Finally, a prototype system is built and implemented on the JADE platform. Experimental trials are simulated to demonstrate the efficiency and effectiveness of the proposed approach. Compared with the best results obtained by MAR2+FIFO for Test Case 1, MAR1+FIFO for Test Case 2, and MAR1+SPT for Test Case 3, the proposed MARS improves makespan by 21.7%, 33.3%, and 0%, critical machine workload by 27.3%, 25.9%, and 0%, and total energy consumption by 31.2%, 50.6%, and 27.2%, respectively, under the real-time shop floor environment.

The contributions of this work can be summarized as follows.

- A new MARS architecture is proposed and implemented on the JADE platform such that an effective real-time scheduling method in the IoT-based manufacturing environment is developed.
- A new multi-agent-based real-time allocation strategy to optimally assign operations to machines is proposed to implement the real-time scheduling in the IoT-based manufacturing environment.
- A bargaining-game-based real-time scheduling method is designed in the RSA to further improve the production efficiency and reduce the processing cost.

Future research is necessary to focus on the improvement of methodology for solving the real-time production scheduling problem with more objectives and practical constraints. In addition, how to integrate the advantages of multi-agent and auto-ID technologies to accomplish integrated process planning and real-time scheduling in a flexible job shop is another issue for future work.

REFERENCES

[1] D. Lei, Y. Zheng, and X. Guo, "A shuffled frog-leaping algorithm for flexible job shop scheduling with the consideration of energy consumption," *Int. J. Prod. Res.*, vol. 55, no. 11, pp. 3126–3140, 2017.

[2] G. Zhang, Y. Zhang, X. Xu, and R. Y. Zhong, "An augmented Lagrangian coordination method for optimal allocation of cloud manufacturing services," *Journal of Manufacturing Systems*, pp. 1–12, 2017.

[3] L.-F. Tung, L. Lin, and R. Nagi, "Multiple-objective scheduling for the hierarchical control of flexible manufacturing systems," *Int. J. Flex. Manuf. Syst.*, vol. 11, no. 4, 1999.

[4] H. Z. Jia, J. Y. H. Fuh, A. Y. C. Nee, and Y. F. Zhang, "Integration of genetic algorithm and Gantt chart for job shop scheduling in distributed manufacturing systems," *Comput. Ind. Eng.*, vol. 53, no. 2, pp. 313–320, 2007.

[5] K. L. Choy, Y. K. Leung, H. K. H. Chow, T. C. Poon, C. K. Kwong, G. T. S. Ho, and S. K. Kwok, "A hybrid scheduling decision support model for minimizing job tardiness in a make-to-order based mould manufacturing environment," *Expert Syst. Appl.*, vol. 38, no. 3, pp. 1931–1941, 2011.

[6] Z. Wu and M. X. Weng, "Multiagent scheduling method with earliness and tardiness objectives in flexible job shops," *IEEE Trans. Syst. Man, Cybern. Part B Cybern.*, vol. 35, no. 2, pp. 293–301, 2005.

[7] N. Q. Wu and M. C. Zhou, "Schedulability analysis and optimal scheduling of dual-arm cluster tools with residency time constraint and activity time variation," *IEEE Transactions on Automation Science and Engineering*, vol. 9, no. 1, 203-209, Jan. 2012.

[8] N. Q. Wu and M. C. Zhou, "Modeling, analysis and control of dual-arm cluster tools with residency time constraint and activity time variation based on Petri nets," *IEEE Transactions on Automation Science and Engineering*, vol. 9, no. 2, 446-454, Apr. 2012.

[9] N. Q. Wu, F. Chu, C. B. Chu, and M. C. Zhou, "Petri net modeling and cycle time analysis of dual-arm cluster tools with wafer revisiting," *IEEE Transactions on Systems*, *Man*, *& Cybernetics*: *Systems*, vol. 43, no. 1, 196-207, Jan. 2013.

[10] N. Q. Wu, M. C. Zhou, L. P. Bai, and Z. W. Li, "Short-term scheduling of crude oil operations in refinery with high fusion point oil and two transportation pipelines," *Enterprise Information Systems*, vol. 10, no. 6, 581-610, May 2016.

[11] L. P. Bai, N. Q. Wu, Z. W. Li, and M. C. Zhou, "Optimal one-wafer cyclic scheduling and buffer space configuration for single-arm multicluster tools with linear topology," *IEEE Transactions on Systems*, *Man*, *& Cybernetics*: *Systems*, vol. 46, no. 10, 1456-1467, Oct. 2016.

[12] F. J. Yang, N. Q. Wu, Y. Qao, M. C. Zhou, and Z. W. Li, "Scheduling of single-arm cluster tools for an atomic layer deposition process with residency time constraints," *IEEE Transactions on Systems*, *Man*, *& Cybernetics*: *Systems*, vol. 47, no. 3, 502-516, Mar. 2017.

[13] Y. Hou, N. Q. Wu, M. C. Zhou, and Z. W. Li, "Pareto-optimization for scheduling of crude oil operations in refinery via genetic algorithm," *IEEE Transactions on Systems*, *Man*, *& Cybernetics*: *Systems*, vol. 47, no. 3, 517-530, Mar. 2017.

[14] S. W. Zhang, N. Q. Wu, Z. W. Li, T. Qu, and C. D. Li, "Petri net-based approach to short-term scheduling of crude oil operations with less tank requirement," *Information Sciences*, vol. 417, 247-261, Nov. 2017.

[15] N. Q. Wu, Z. W. Li, and T. Qu, "Energy efficiency optimization in scheduling crude oil operations of refinery based on linear programming," *Journal of Cleaner Production*, vol. 166, 49-57, Nov. 2017.

[16] Q. H. Zhu, M. C. Zhou, Y. Qiao, and N. Q. Wu, "Petri net modeling and scheduling of a close-down process for time-constrained single-arm cluster tools," *IEEE Transactions on Systems*, *Man*, *and Cybernetics*: *Systems*, vol. 48, no. 3, 389-400, Mar. 2018.

[17] Y. Qiao, N. Q. Wu, F. J. Yang, M. C. Zhou, and Q. H. Zhu, "Wafer sojourn time fluctuation analysis of time-constrained dual-arm cluster tools with wafer revisiting and activity time variation," *IEEE Transactions on Systems*, *Man*, *and Cybernetics*: *Systems*, vol. 48, no. 4, 622-636, Apr. 2018.

[18] L. Shen, S. Dauzère-Pérès, and J. S. Neufeld, "Solving the flexible job shop scheduling problem with sequence-dependent setup times," *Eur. J. Oper. Res.*, vol. 265, no. 2, pp. 503–516, 2018.

[19] M. Gen, W. Zhang, L. Lin, and Y. S. Yun, "Recent advances in hybrid evolutionary algorithms for multiobjective manufacturing scheduling," *Comput. Ind. Eng.*, vol. 112, pp. 616–633, 2017.

[20] M. Merdan, T. Moser, W. Sunindyo, S. Biffl, and P. Vrba, "Workflow scheduling using multi-agent systems in a dynamically changing environment," *J. Simul.*, vol. 7, no. 3, pp. 144–158, 2013.

[21] S. Zhang and T. N. Wong, "Flexible job-shop scheduling/rescheduling in dynamic environment: a hybrid MAS/ACO approach," *Int. J. Prod. Res.*, vol. 55, no. 11, pp. 3173–3196, 2017.

[22] R. Erol, C. Sahin, A. Baykasoglu, and V. Kaplanoglu, "A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles in manufacturing systems," *Appl. Soft Comput. J.*, vol. 12, no. 6, pp. 1720–1732, 2012.

[23] J. Yuan, "Multi-agent scheduling on a single machine with a fixed number of competing agents to minimize the weighted sum of number of tardy jobs and makespans," *J. Comb. Optim.*, vol. 34, no. 2, pp. 433–440, 2017.

[24] C. Sahin, M. Demirtas, R. Erol, A. Baykasoğlu, and V. Kaplanoğlu, "A multi-agent based approach to dynamic scheduling with flexible processing capabilities," *J. Intell. Manuf.*, 2015.

[25] Y. Zhang, J. Wang, and Y. Liu, "Game theory based real-time multi-objective flexible job shop scheduling considering environmental impact," *J. Clean. Prod.*, vol. 167, pp. 665–679, 2017.

[26] Y. Zhang, S. Ma, H. Yang, J. Lv, and Y. Liu, "A big data driven analytical framework for energy-intensive manufacturing industries," *J. Clean. Prod.*, vol. 197, pp. 57–72, 2018.

[27] Y. Zhang, Z. Guo, J. Lv, and Y. Liu, "A Framework for Smart Production-Logistics Systems based on CPS and Industrial IoT," *IEEE Transactions on Industrial Informatics*, 2018.

[28] Y. Zhang, G. Zhang, J. Wang, S. Sun, S. Si, and T. Yang, "Real-time information capturing and integration framework of the internet of manufacturing things," *Int. J. Comput. Integr. Manuf.*, vol. 3052, no. December, pp. 1–12, Aug. 2014.

[29] S. Liu, G. Zhang, and L. Wang, "IoT-enabled Dynamic Optimisation for Sustainable Reverse Logistics," *Procedia CIRP*, vol. 69, no. April, pp. 662–667, 2018.

[30] Y. Zhang, S. Liu, Y. Liu, H. Yang, M. Li, D. Huisingh, and L. Wang, "The 'Internet of Things' enabled real-time scheduling for remanufacturing of automobile engines," *J. Clean. Prod.*, vol. 185, pp. 562–575, 2018.

[31] Y. Zhang, S. Ren, Y. Liu, T. Sakao, and D. Huisingh, "A framework for Big Data driven product lifecycle management," *J. Clean. Prod.*, vol. 159, pp. 229–240, 2017.

[32] X. Z. Ren Shan, "A framework for shopfloor material delivery based on real-time manufacturing big data," *J. Ambient Intell. Humaniz. Comput.*, 2018.

[33] Y. Zhang, G. Zhang, W. Du, J. Wang, E. Ali, and S. Sun, "An optimization method for shopfloor material handling based on real-time and multi-source manufacturing data," *Int. J. Prod. Econ.*, vol. 165, pp. 282–292, 2015.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2871346, IEEE Internet of Things Journal

[34] 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.

[35] F. Zhao, J. Wang, J. Wang, and J. Jonrinaldi, "A dynamic rescheduling model with multi-agent system and its solution method," *Stroj. Vestnik/Journal Mech. Eng.*, vol. 58, no. 2, pp. 81–92, 2012.

[36] X. Chen, H. Wen Lin, and T. Murata, "Composite dispatching rule design for dynamic scheduling with customer-oriented production priority control," *IEEJ Trans. Electr. Electron. Eng.*, vol. 7, no. 1, pp. 53–61, 2012.

[37] T. Ning, M. Huang, X. Liang, and H. Jin, "A novel dynamic scheduling strategy for solving flexible job-shop problems," *J. Ambient Intell. Humaniz. Comput.*, vol. 7, no. 5, pp. 721–729, 2016.

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

[39] R. Rangsaritratsamee, W. G. Ferrell, and M. B. Kurz, "Dynamic rescheduling that simultaneously considers efficiency and stability," *Comput. Ind. Eng.*, vol. 46, no. 1, pp. 1–15, 2004.

[40] N. Kumar, M. K. Tiwari, and F. T. S. Chan, *Development of a hybrid negotiation scheme for multi-agent manufacturing systems*, vol. 46, no. 3. 2008.

[41] H. Tehrani Nik Nejad, N. Sugimura, K. Iwamura, and Y. Tanimizu, "Multi agent architecture for dynamic incremental process planning in the flexible manufacturing system," *J. Intell. Manuf.*, vol. 21, no. 4, pp. 487–499, 2010.

[42] P. I. Cowling, D. Ouelhadj, and S. Petrovic, "Dynamic scheduling of steel casting and milling using multi-agents," *Prod. Plan. Control*, vol. 15, no. 2, pp. 178–188, 2004.

[43] M. J.-P. Shaw, "Distributed Planning in Cellular Flexible Manufacturing Systems," *INFOR*, vol. 25, no. 1, pp. 13–25, 1987.

[44] H. V. D. Parunak, "Manufacturing experience with the contract net," in *Distributed artificial intelligence*, 1987, pp. 285–310.

[45] F.-S. Hsieh, "A hybrid and scalable multi-agent approach for patient scheduling based on Petri net models," *Appl. Intell.*, vol. 47, no. 4, pp. 1068–1086, 2017.

[46] P. Perez-Gonzalez and J. M. Framinan, "A common framework and taxonomy for multicriteria scheduling problems with interfering and competing jobs: Multi-agent scheduling problems," *European Journal of Operational Research*, vol. 235, no. 1. pp. 1–16, 2014.

[47] M. M. Savino, A. Mazza, and G. Neubert, "Agent-based flow-shop modelling in dynamic environment," *Prod. Plan. Control*, vol. 25, no. 2, pp. 110–122, 2014.

[48] M. M. Savino, A. Brun, and A. Mazza, "Dynamic workforce allocation in a constrained flow shop with multi-agent system," *Comput. Ind.*, vol. 65, no. 6, pp. 967–975, 2014.

[49] A. V. Barenji, R. V. Barenji, D. Roudi, and M. Hashemipour, "A dynamic multi-agent-based scheduling approach for SMEs," *Int. J. Adv. Manuf. Technol.*, vol. 89, no. 9–12, pp. 3123–3137, 2017.

[50] A. Kouider and B. Bouzouia, "Multi-agent job shop scheduling system based on co-operative approach of idle time minimisation," *Int. J. Prod. Res.*, vol. 50, no. 2, pp. 409–424, 2012.

[51] X. Yu and B. Ram, "Bio-inspired scheduling for dynamic job shops with flexible routing and sequence-dependent setups," *Int. J. Prod. Res.*, vol. 44, no. 22, pp. 4793–4813, 2006.

[52] H. T. N. Nejad, N. Sugimura, and K. Iwamura, "Agent-based dynamic integrated process planning and scheduling in flexible manufacturing systems," *Int. J. Prod. Res.*, vol. 49, no. 5, pp. 1373–1389, 2011.

[53] N. Mishra, A. Singh, S. Kumari, K. Govindan, and S. I. Ali, "Cloud-based multi-agent architecture for effective planning and scheduling of distributed manufacturing," *Int. J. Prod. Res.*, vol. 54, no. 23, pp. 7115–7128, 2016.

[54] K. Y. Chen and C. J. Chen, "Applying multi-agent technique in multi-section flexible manufacturing system," in *Expert Systems with Applications*, 2010, vol. 37, no. 11, pp. 7310–7318.

[55] R. Erol, C. Sahin, A. Baykasoglu, and V. Kaplanoglu, "A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles in manufacturing systems," *Appl. Soft Comput. J.*, vol. 12, no. 6, pp. 1720–1732, 2012.

[56] Y. Zhang, Z. Zhu, and J. Lv, "CPS-Based smart control model for shopfloor material handling," *IEEE Trans. Ind. Informatics*, vol. 3203, pp. 1–1, 2017.

[57] S. jin Wang, L. feng Xi, and B. hai Zhou, "FBS-enhanced agent-based dynamic scheduling in FMS," *Eng. Appl. Artif. Intell.*, vol. 21, no. 4, pp. 644–657, 2008.

[58] E. Rowland, *Theory of Games and Economic Behavior*, vol. 157, no. 3981. 1946.

[59] X. Sun, Y. Liu, J. Li, J. Zhu, H. Chen, and X. Liu, "Feature evaluation and selection with cooperative game theory," *Pattern Recognition*, vol. 45, no. 8. pp. 2992–3002, 2012.

[60] P. Argoneto and P. Renna, "Supporting capacity sharing in the cloud manufacturing environment based on game theory and fuzzy logic," *Enterp. Inf. Syst.*, vol. 10, no. 2, pp. 193–210, 2016.

[61] Y. Liu, L. Zhang, F. Tao, and L. Wang, "Resource service sharing in cloud manufacturing based on the Gale–Shapley algorithm: advantages and challenge," *Int. J. Comput. Integr. Manuf.*, vol. 30, no. 4–5, pp. 420–432, 2017.

[62] G. Zhou, Z. Xiao, P. Jiang, and G. Q. Huang, "A game-theoretic approach to generating optimal process plans of multiple jobs in networked manufacturing," *Int. J. Comput. Integr. Manuf.*, vol. 23, no. 12, pp. 1118–1132, 2010.

[63] P. Calleja, A. Est??vez-Fern??ndez, P. Borm, and H. Hamers, "Job scheduling, cooperation, and control," *Oper. Res. Lett.*, vol. 34, no. 1, pp. 22–28, 2006.

[64] Z. Han, Y. Zhu, X. Ma, and Z. Chen, "Multiple rules with game theoretic analysis for flexible flow shop scheduling problem with component altering times," *Int. J. Model. Identif. Control*, vol. 26, no. 1, p. 1, 2016.

[65] G. Zhou, P. Jiang, and G. Q. Huang, "A game-theory approach for job scheduling in networked manufacturing," *Int. J. Adv. Manuf. Technol.*, vol. 41, no. 9–10, pp. 972–985, 2009.

[66] D. Diepl and J. Reaidy, "Scheduling agents in a distributed flexible manufacturing system," in *IEEE International Symposium on Industrial Electronics*, 2004, vol. 1, pp. 739–744.

[67] J. Reaidy, P. Massotte, and D. Diep, "Comparison of negotiation protocols in dynamic agent-based manufacturing systems," in *International Journal of Production Economics*, 2006, vol. 99, no. 1–2, pp. 117–130.

[68] A. Agnetis, G. Nicosia, A. Pacifici, and U. Pferschy, "Scheduling two agent task chains with a central selection mechanism," *J. Sched.*, vol. 18, no. 3, pp. 243–261, 2015.

[69] J. Watson, *Strategy*. 2013.

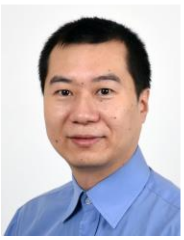
[70] I. Kacem, S. Hammadi, and P. Borne, "Pareto-optimality approach for flexible job-shop scheduling problems: Hybridization of evolutionary algorithms and fuzzy logic," *Math. Comput. Simul.*, vol. 60, no. 3–5, pp. 245–276, Sep. 2002.

[71] Y. He, Y. Li, T. Wu, and J. W. Sutherland, "An energy-responsive optimization method for machine tool selection and operation sequence in flexible machining job shops," *J. Clean. Prod.*, vol. 87, no. C, pp. 245–254, 2015.

Jin Wang received the B.S. and M.S. degrees in Mechanical Engineering both from Xi'an University of Science and Technology, Xi'an, China, in 2007 and 2010 respectively.

He is currently working toward the Ph.D. degree in Mechanical Engineering, Northwestern Polytechnical University, Xi'an, China. His research interest is real-time data-based production management.

Yingfeng Zhang (M'17) received the B.S., M.S., and Ph.D. degrees in mechanical engineering from Xi'an Jiaotong University, Xi'an, China, in 1999, 2002, and 2005, respectively. He is currently a Professor with the Department of Industrial Engineering, Northwestern Polytechnical University, Xi'an, China. His research interests include framework and core models of future intelligent industrial system. He has published over 50 papers in various international journals. He has organized three special issues of International Journal of Cleaner Production, Journal of Intelligent Manufacturing, and International Journal of Computer Integrated Manufacturing. His publications are cited 2000+ by Google scholar.



Yang Liu received his M.Sc. (Tech.) in Telecommunications Engineering and D.Sc. (Tech.) in Industrial Management from the University of Vaasa, Finland, in 2005 and 2010, respectively. He currently works as a tenured Associate Professor and Doctoral Supervisor in the Department of Management and Engineering at Linköping University (Sweden), a part-time faculty in the Department of Production at University of Vaasa (Finland), and a Chair Professor at Jinan University (China). He is appointed Adjunct/Visiting Professor in multiple other universities. His main research interests include smart manufacturing, product service innovation, decision support system, sustainable competitive advantage; control systems, autonomous robots, signal processing and pattern recognition.

NaiQi Wu (M'04–SM'05) received his B. S. Degree in Electrical Engineering from Anhui University of Technology, Huainan, China, in 1982, the M. S. and Ph. D. Degrees in Systems Engineering both from Xi'an Jiaotong University, Xi'an, China in 1985 and 1988, respectively. From1988 to 1995, he was with Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China, and from 1995 to 1998, with Shantou University, Shantou, China. He moved to Guangdong University of Technology, Guangzhou, China in 1998. He joined Macau University of Science and Technology, Taipa, Macau in 2013. He is currently a Professor at the Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau. His research interests include production planning and scheduling, manufacturing system modeling and control, discrete event systems, Petri net theory and applications, intelligent transportation systems, and energy systems. He is the author or coauthor of one book, five book chapters, and 140+ peer-reviewed journal articles. Dr. Wu was an associate editor of the IEEE Transactions on Systems, Man, & Cybernetics, Part C, IEEE Transactions on Automation Science and Engineering, IEEE Transactions on Systems, Man, & Cybernetics: Systems, and editor in chief of Industrial Engineering Journal, and is an associate editor of Information Sciences and IEEE/CAA Journal of Automatica Sinica.