Infinitely repeated game based real-time scheduling for low-carbon flexible job shop considering multi-time periods

Infinitely repeated game based real-time scheduling for low-carbon flexible job shop considering multi-time periods

Jin Wang [a,b] *, JiahaoYang [a] ,Yingfeng Zhang [a] *, Shan Ren [a,b] *, Yang Liu [c,d] *

[a] Key Laboratory of Contemporary Design and Integrated Manufacturing Technology, Ministry of Education, Northwestern Polytechnical University, Shaanxi, 710072, PR China

[b] School of Modern Posts, Xi'an University of Posts and Telecommunications, Shaanxi, 710061, PR China

[c] Department of Management and Engineering, Linköping University, SE-581 83 Linköping, Sweden

[d] Department of Production, University of Vaasa, 65200 Vaasa, Finland

* Corresponding Author: zhangyf@nwpu.edu.cn (Y. Zhang), wj19852004@qq.com (J. Wang) renshan@mail.nwpu.edu.cn (S. Ren), yang.liu@liu.se (Y. Liu)

Abstract: Production scheduling has great significance for optimizing tasks distribution, reducing energy consumption and mitigating environmental degradation. Currently, the research of production scheduling considering energy consumption mainly focuses on the traditional manufacturing workshop. With the wide application of the Internet of Things (IoT) technology, the real-time data of manufacturing resources and production processes can be retrieved easily. These manufacturing data can provide opportunities for manufacturing enterprises to reduce energy consumption and enhance production efficiency. To achieve these targets, a multi-period production planning based real-time scheduling (MPPRS) approach for the IoT-enabled low-carbon flexible job shop (LFJS) is presented in this study to carry out real-time scheduling based on the real-time manufacturing data. Then, the mathematical models of real-time scheduling are established to achieve production efficiency improvement and energy consumption reduction. To obtain a feasible solution, an infinitely repeated game optimization approach is used. Finally, a case study is implemented to analyse and discuss the effectiveness of the proposed method. The results show that in general, the proposed method can achieve better results than the existing dynamic scheduling methods.

Keywords: Energy consumption, Real-time scheduling, Flexible job shop, Infinitely repeated game

1. Introduction

Currently, with global warming, the manufacturing industry is facing huge challenges in environmental protection and cleaner production (Y. Zhang et al., 2018; Ma et al., 2019). As pointed by Wu et al. (2018), today, the industrial enterprise accounts for about half of the world's energy consumption, which has almost doubled in the last sixty years. The rising energy consumption has brought enormous environmental pressures, for instance, greenhouse gas emissions contribute to global warming (Z. Zhang et al., 2016). (Hondo, 2005) studied the amount of greenhouse gases (CO_2 and CH_4) emitted to generate per kWh of electricity. Therefore, improving energy efficiency in industrial enterprises to reduce carbon emissions has become an

urgent problem (Cassettari et al., 2017). Over the past ten years, reducing energy consumption has been achieved mainly by improving existing machines and processes. In recent years, new operation strategies for reducing energy consumption have raised concerns. In particular, researchers have realized that appropriate scheduling mechanisms and methods could play a crucial role in energy-saving during the production execution stage (Dai et al., 2015; Yan et al., 2016; Raileanu et al., 2017). Moreover, production scheduling has attracted much attention (Lei, 2008; Renna, 2010), especially for flexible job shop scheduling (FJSS) problem (Jula and Kones, 2013; Nouiri et al., 2015). However, most research on this topic assumed that the production environment is static, thereby no unexpected disruption occurred during the production processes.

In the real production workshop, it is quite common for the production processes to encounter unexpected disruption such as rush order, machine breakdown and worker absenteeism (Nguyen et al., 2015). In such circumstances, the predetermined schedules lose their optimality or even become infeasible to be executed (Zhang and Wong, 2017). This means that static scheduling has little contribution to reducing energy consumption in a real production environment. Thus, the dynamic scheduling problem has attracted some researchers' attention in recent years (Ning et al., 2016; Wang et al., 2017). Meanwhile, some approaches and algorithms have been proposed, such as swarm intelligence approach (Rossi, 2014), gravitational emulation local search algorithm (Hosseinabadi et al., 2015), and the approach of integrating genetic algorithm with particle swarm optimization (Wang et al., 2018). However, the above-mentioned studies focus mainly on the dynamic scheduling problem in a traditional production workshop.

In recent years, the development of information and communications technology provide technical support for manufacturing enterprise to implement production management based on real-time data (Zhang et al., 2017; Zhang et al., 2019; Wang et al., 2020). With these technologies, manufacturing enterprises can carry out real-time traceability and visibility of manufacturing resources and improve workshop scheduling performance (Qian et al., 2019; Liu et al., 2019). Today, by applying the IoT technology to manufacturing fields (Guo et al., 2015; Qian et al., 2020), the real-time manufacturing data of workshop has become more accessible to create a big data environment for manufacturing (Y. Zhang et al., 2018b; Huang et al., 2019). In this environment, the traditional scheduling methods could be challenged in dynamic FJSS problem. Recently, Ren et al. (2018) introduced the development trends of a future production system in a big data environment. They pointed out that real-time data of workshop can make dynamic scheduling more efficient.

Despite these researchers have made significant progress, the following research questions should be solved in applying FJSS methods to the real-world production workshop. They are listed as follows.

How to design a real-time scheduling method to implement production optimization for the IoT-enabled LFJS by the infinitely repeated game? In general, there are two kinds of mainly dynamic scheduling methods: periodic and event-driven dynamic scheduling. For the periodic

dynamic scheduling, a new schedule is generated after a certain scheduling interval. For the event-driven dynamic method, the dynamic scheduling is performed when the previous schedule is modified to accommodate the new manufacturing environment. However, the scheduling strategies derive from these two methods have some problems. For instance, the new schedule may be completely different from the original one, meaning that tasks that were not processed in the original schedule may be processed earlier or delay (Rangsaritratsamee et al., 2004). It has a serious effect on other production activities planning related to the original schedule and reduces the stability of the production scheduling system (Shen and Yao, 2015). However, in an infinitely repeated game based real-time scheduling approach, during the real-time scheduling stage, each machine can automatically send its real-time status and request of the operations when it is idle at each time. The operations continually interact with the machines and operators according to their real-time status. Each time, only one optimal operation is assigned to the requested machine. When the machine finishes the assigned operation, it automatically sends its current status and requests the operations again until all the operations are finished. Since the operation allocation is real-time data-driven and the infinitely repeated game based real-time scheduling approach is only started for the idle machine, the efficiency can be largely improved. Thus, the infinitely repeated game based real-time scheduling approach needs to be developed to reduce the adverse impacts of the unpredictable exceptions in the IoT-enabled LFJS.

How to design a production operation allocation policy to enhance the production efficiency and alleviate the environmental pollution of the IoT-enabled LFJS? In a production environment, production planning is very important to improve the processing efficiency of the workshop (Davis and Thompson, 1993). Production scheduling is also a key factor affecting production efficiency (Luh et al., 1997). Their results can strongly influence the development of corporate profits, as well as the efficiency of utilization of resources (Chen, 2016). Recently, many studies have been devoted to exploring the production planning and scheduling problem. However, there are few studies on simultaneously optimizing the production planning and scheduling for a flexible production environment. Although some literature considered the integration of the production planning and scheduling problem, these studies only generate a production planning at the initial time and do not consider the impact of unexpected events on production planning. Moreover, the research on the integration of production planning and scheduling based on real-time data is almost vacant. Therefore, a new MPPRS method should be proposed to enhance production efficiency and energy efficiency in the IoT-enabled LFJS.

To address these problems, in this study, an MPPRS method based on the infinitely repeated game for IoT-enabled LFJS is proposed to give a new production scheduling idea. The proposed method integrates three important features. First is the application of IoT technology to the real-time scheduling problem. In the IoT-enabled production workshop, the manufacturing resources can interact with each other, and the real-time status of manufacturing resources can be monitored. The results of real-time scheduling can be obtained based on their real-time status. Second is the

developed MPPRS method to enhance production efficiency and reduce environmental pollution. The production horizon is divided into multiple periods. At the start of each period, the production planning is generated, and then the real-time scheduling can be realized to satisfy the requirement from the production planning in each period. Third is the proposed infinitely repeated game optimization method for MPPRS. It is used to assign the operations to the corresponding machines depending on the machine's real-time status. Therefore, the designed MPPRS based on real-time manufacturing data provides a new production scheduling paradigm to further alleviate environment pollution and improve production efficiency for the IoT-enabled LFJS.

The remainder of the study is arranged as follows. A literature review is conducted in Section 2. The implementation processes of MPPRS are illustrated in Section 3. In Section 4, the MPPRS model is proposed. Section 5 develops the infinitely repeated game optimization method for MPPRS. A case study is used to demonstrate the efficiency of the designed MPPRS in Section 6. Section 7 gives the conclusions and future works.

2. Literature review

Related literature is reviewed under two parts: (1) the real-time scheduling and (2) the game theory based production scheduling considering energy consumption.

2.1 Real-time scheduling

The dynamic scheduling problem occurs when exceptional events such as rush order and cancellation of jobs are considered in a static production scheduling system. Holloway and Nelson (1974) were the first to address the dynamic scheduling problem. They pointed out that the periodic policy is useful in a dynamic job shop problem. Subsequently, Muhleman et al. (1982) further studied the application of the periodic scheduling strategy in a dynamic random job shop system. Over the next few years, Church and Uzsoy (1992) developed periodic and event-driven rescheduling methods in a single machine scheduling system with rush order. Then, a simple heuristic dispatching rule, called a shift from standard rules, was designed by Pierreval and Mebarki (1997). The same year, Kouiss K and Pierreval H (1997) first proposed a scheduling strategy based on a multi-agent for a dynamic job shop. In recent decades, more and more dynamic scheduling methods have been put forward to cope with the exceptional events in the dynamic scheduling system (Aytug et al., 2005; Ouelhadj and Petrovic, 2009). In the literature, the dynamic scheduling method can be categorized into three groups: (1) artificial intelligence (AI) and knowledge-based approaches, such as genetic algorithm (GA) (Zambrano Rey et al., 2014; Kundakci and Kulak, 2016), neural networks (Tang et al., 2005; Araz and Salum, 2010), expert systems (Vieira et al., 2000; Umar et al., 2015), and fuzzy logic (Zhao et al., 2010; Xanthopoulos et al., 2013); (2) simulation-based approaches, like the priority dispatching rules (Lu and Romanowski, 2013; Sharma and Jain, 2016); (3) multi-agent-based approaches (Wang et al., 2008; Sahin et al., 2017). However, these traditional dynamic scheduling methods are intrinsically inflexible, slower responsive to exceptional events and cannot satisfy the needs of unanticipated

real-time situations.

With the wide application of computer and communication technologies, some new management manners have been proposed to optimize the production process and to enhance the efficiency of production scheduling. Especially, over the past decade, many enterprises have used IoT technology to support workshop management, where the manufacturing resources can interact with each other dynamically. Consequently, a new manufacturing environment namely the Internet of Manufacturing Things (IoMT) is built. Under this IoMT environment, large amounts of manufacturing data in the workshop are constantly exchanged and interplayed. As a result, the traditional dynamic scheduling methods may no longer be appropriate in the IoMT environment.

In addition, the existing researches on production planning based real-time scheduling for manufacturing workshop are quite limited. There are only a few papers address this area by a rigorous literature search. In this literature, Luo et al. (2015) present a multi-period hierarchical scheduling mechanism to optimize the objectives of production planning and real-time scheduling respectively for hybrid flow shop. In follow-up work, Shukla et al. (2018) developed an agent-based architecture that enables the integration of production planning and scheduling for a job shop. However, these works seldom focus on the FJSS problem considering the real-time manufacturing data.

2.2 Game theory based production scheduling considering energy consumption

Game theory is a popular approach to deal with the optimal decision problem (Li et al., 2012). It is mainly used in economics, political science, communication, and psychology, as well as logic and biology. In recent decades, some researchers extended the application of game theory to solve the problem from different areas of engineering (Miyamoto et al., 2008; Hu and Rao, 2009). At present, more and more researchers have begun to pay close attention to game theory and use it to solve the production scheduling problem. The game theory can be divided into two categories: cooperative game and non-cooperative game (Sun et al., 2014). For the cooperative game, Calleja et al. (2006) considered one machine sequencing situations using the cooperative game. Arasteh et al. (2014) developed the application of cooperative game theory in scheduling optimization. Han et al. (2016) used a cooperative game to study the flexible flow shop scheduling problem (FFSP). For the non-cooperative game, an agent-based production scheduling method is presented to deal with the flexible flow shop scheduling problem using incorporating game theoretic (Babayan and He, 2004). Zhou et al. (2009) presented a production scheduling method in a networked production workshop using the non-cooperative game. Manupati et al. (2012) proposed a scheme to generate optimal production planning and scheduling for network manufacturing workshop. Agnetis et al. (2015) developed a minimax strategy to deal with a scheduling problem using game theory.

These works have put forward many new ideas for the application of game theory in a production scheduling problem. However, from the literature, we have noticed that the above works mainly focus on time-based performances, especially the minimization of makespan. Little

attention has been paid to the energy consumption of production. In recent years, given concerns related to climate change, many researchers turn their attention to the energy consumption of production scheduling, especially the LFJS (Mokhtari and Hasani, 2017). For example, Zhang et al. (2016) proposed an innovative approach to study the dynamic scheduling problem in FMS, taking the objectives of minimum or maximum energy consumption into account. Yin et al. (2017) presented a low-carbon mathematical scheduling model for the flexible job shop environment that optimizes productivity, energy efficiency and noise reduction. Lei et al. (2017) considered the total energy consumption and workload balance and proposed a shuffled frog leaping algorithm to solve the FJSS problem. Wu and Sun (2018) applied a non-dominated GA for the FJSS problem with energy-saving measures. Meng et al. (2019) proposed several mixed-integer linear programming models of higher efficiency to solve the LFJS problem. However, the above works did not use game theory to solve the LFJS problem. The author only found a few studies after a systematic literature search. For example, Zhang et al. (2017b) proposed a two-layer scheduling method based on dynamic game theory to reduce energy consumption in a flexible job shop. Then, Wang et al. (2018) proposed a multi-agent-based real-time scheduling (MARS) method to optimize energy objective using a bargaining game. However, in the above papers, players of the game only play once. Thus, players focus too much on the short-term interests of individuals to make strategic choices, and the payoff of each player may have less benefit than the infinitely repeated game solution, meaning that the solutions of these studies may not be optimal. To address these challenges, this study proposed an MPPRS method to realize real-time data-driven scheduling for IoT-enabled LFJS by using the infinitely repeated game.

3. The implementation processes of MPPRS

This section mainly discusses the implementation processes of MPPRS in the IoT-enabled LFJS. The objective of the proposed MPPRS is to enhance the real-time interaction of distributed manufacturing resources by applying IoT technology to LFJS, and then the real-time optimization of manufacturing operations can be achieved in a sensible manufacturing environment. The sensible manufacturing environment is enabled by RFID facilities. Each machine is equipped with a reader and some antennas to connect them. These antennas can capture the real-time data from kinds of RFID tags that are attached to manufacturing resources (e.g. machines, key WIP (work-in-progress) items, operators, etc.) Thus, the machines can actively sense the manufacturing environment and dynamically interact with each other. The real-time data of the production workshop could be tracked and traced.

The implementation processes of MPPRS are shown in Fig.1, which includes two layers, namely the production planning layer and real-time scheduling layer. The production planning layer divides the production horizon into multiple periods T and makes one decision at the beginning of every period T. The real-time scheduling layer divides one period T into multiple time t, and the real-time scheduling results can be worked out at each time t according to the real-

time manufacturing data of the workshop. In this study, the time unit of T is calculated by equation (1).

$$T = [\frac{\max C_i^c}{n'}] \tag{1}$$

The C_i^c denotes the planned completion time of task i in the first production planning. The n' denotes the parameter, which is used to adjust T. If n' is larger, it means that T is smaller, and the frequency of production planning generation is higher. Therefore, the system can respond to the influence of exceptional events on production planning in time, which means that the system requires higher rapidity and stability. On the other hand, if n' is smaller, it means T is larger, and the frequency of production planning generation is lower. Therefore, the response time of production planning to exceptional events is longer, which means that the system requires lower rapidity and stability.

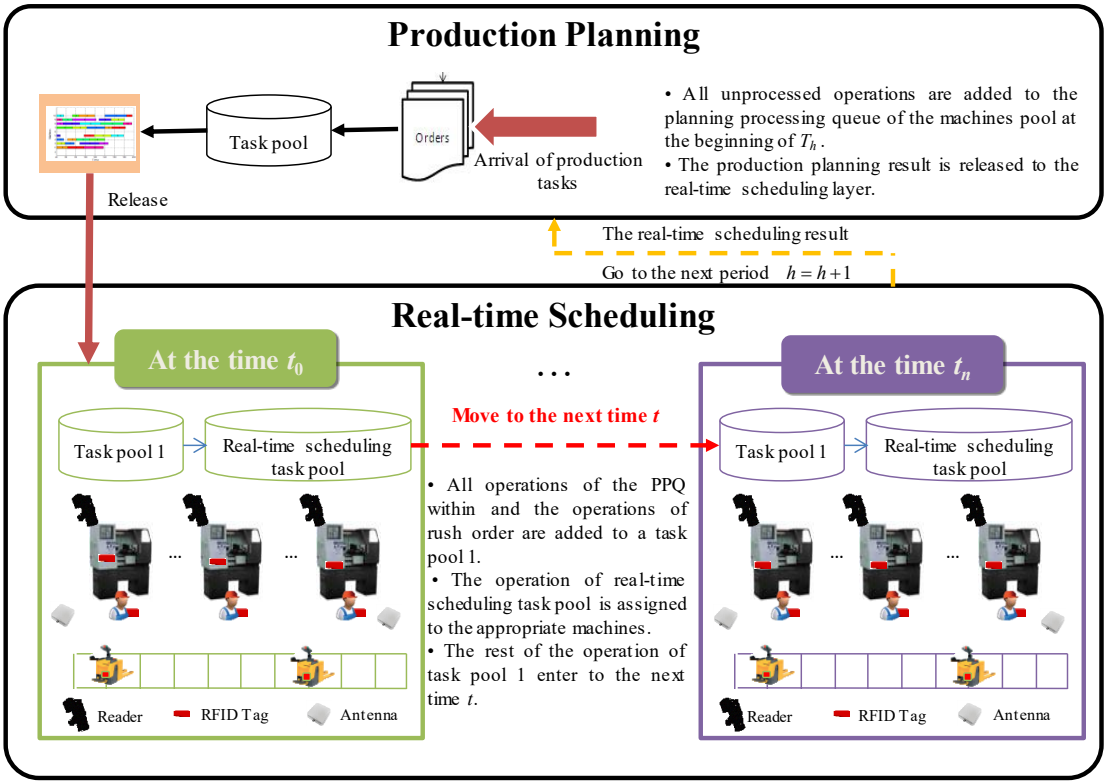

Fig.1. The implementation processes of MPPRS

The implementation processes of MPPRS are shown as follows:

Step 1: at the beginning of T_h, the real-time data of the new arriving production tasks and the unallocated operations can be obtained by the production planning layer. Here, T_h denotes the h^{th} period. Then, the first unallocated operations of each task are put into a task pool (TP). Next, only one operation of the TP is added to the planning processing queue (PPQ) of the machines by using the infinitely repeated game. Repeat the above processes until all operations are put into the PPQ of the machines. At last, the production planning result is released to the real-time scheduling layer.

Step 2: at time t_0 ($t_0 \in T_h$), all operations of the PPQ within T_h and the operations of rush order are added to a task pool 1 (TP1). Then, according to the production planning result, the optional operations of the TP1 at time t_0 are added to a real-time scheduling task pool (RSTP). Next, each machine constantly interacts with the operations and competes to process these operations. At last, the operations are assigned to the appropriate machine by using the infinitely repeated game. The rest of the operations of the TP1 enter the next time t_1. Repeat the above process until the last time t_n ($t_n \in T_h$), the operations are assigned to the most appropriate machines.

Step 3: At the beginning of T_{h+1} the real-time scheduling results within T_h is informed to the production planning layer. Repeat the above steps until the assignment is complete.

4. The MPPRS model

4.1 Problem statement

The MPPRS could be formulated as follows in this study. The workshop has a total of m machines and n tasks. Each task i consists of n_i operations. Each operation of task i has to be processed on machine M_k. The MPPRS is needed to determine that which operation should be processed in which machine and the start process time of the operation so as to satisfy the objective of production planning layer at the beginning of every period T and the objective of real-time scheduling layer at each time t based on the real-time manufacturing data. The notations used in this study are summarized in Table 1.

Table 1 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
$M = \{M_1, M_2, ..., M_m\}$	The set of machines
M_{ij}	The set of machines for processing the operation 'j' of task 'i'
M_{idle}^t	The idle machine at time t
O_{ij}	The j^{th} operation of task i
O_{option}^t	The set of the optional operations at time t
C_{ij}	The completion time of O_{ij}
T_h	The h^{th} period $h = 1, 2, 3, \cdots, n'$
W_k	The workload of M_k
W_M	The critical machine workload, which is the machine with the heaviest workload
W_{zwl}	The total workload of machines
E	The total energy consumption of production
x_{ijk}	1,if M_k is selected for the O_{ij} ; 0,otherwise
P_{idle}^k	The idle power of M_k (kW)
$P_{cutting}^k$	The cutting power of M_k (kW)
t_{idle}^k	The total idle time of M_k

t_{ijkc}	The cutting time of O_{ij} operated on M_k
t_{ijkt}	The tool change time of O_{ij} operated on M_k
g_i	The maximum feasible payoff for player *i* of the stage game

Because the complexity of MPPRS is so high, in the study, some assumptions are made as follows.

(1) Once an operation is started on a machine, it cannot be interrupted unless the machine breaks down.

(2) Each machine can start to process an operation only after the previous operation is completed.

(3) The tool change time is not zero for two consecutive operations.

(4) The task transportation time among machines is not considered.

4.2 Mathematical model

4.2.1 Objective functions of production planning layer

In the production planning layer, the workshop manager makes upper-level production planning. The aim of production planning is to improve the production efficiency and critical machine workload in every period *T*.

Objectives:

(1) Minimizing the makespan:

$$\text{Min} \quad f_1^p = Makespan = \max C_{ij} \tag{2}$$

(2) Minimizing the critical machine workload:

$$\text{Min} \quad f_2^p = W_M = \text{Max}\{W_k\} \tag{3}$$

$$W_k = \sum_{i=1}^{n} \sum_{j=1}^{n_i} [(t_{ijkc} + t_{ijkt}) x_{ijk}] \tag{4}$$

Subject to:

$$C_{ij} \geq 0, \ \forall i, j \qquad C_{ij} - C_{i,j-1} \geq (t_{ijkc} + t_{ijkt}) x_{ijk}, \qquad j = 2, 3, \cdots, n_i \tag{5}$$

$$\sum_{k \in M_{ij}} x_{ijk} = 1 \tag{6}$$

Equation (2) guarantees the minimization of maximal completion time of all the assigning operations. Equation (3) ensures the minimization of the critical machine workload. Equation (4) defines the workload of machine M_k. Inequity (5) guarantees that the operations precedence constraints of tasks. Equation (6) indicates that an operation can only be processed by one machine.

4.2.2 Objective functions of real-time scheduling layer

In the real-time scheduling layer, the scheduling manager assigns all the tasks from upper-level production planning and rush orders during period *T*. The objective is to minimize the makespan, the total workload of machines and the total energy consumption of production.

Objectives:

(1) Minimizing the makespan:

$$\text{Min } f_1^r = Makespan = \max C_{ij} \tag{7}$$

(2) Minimizing the total workload of machines:

$$\text{Min } f_2^r = W_{zwl} = \sum_{k=1}^{m} W_k \tag{8}$$

(3) Minimizing the total energy consumption of production:

$$\text{Min } f_3^r = E = \sum_{k=1}^{m}(P_{idle}^k \cdot t_{idle}^k) + \sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n_i}(P_{cutting}^k \cdot t_{ijkc} \cdot x_{ijk}) + \sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n_i}(P_{idle}^k \cdot t_{ijkt} \cdot x_{ijk}) \tag{9}$$

The equations (7)–(9) are the objective functions. In this study, the real-time scheduling layer considers all these three objectives.

5. Infinitely repeated game optimization method for MPPRS

In this section, the infinitely repeated game based MPPRS method for manufacturing tasks is designed. Through the infinitely repeated game method, the optimal combination of all operations for each machine can be acquired. In view of the huge manufacturing operations in the production enterprises, the production planning layer and real-time scheduling layer are designed respectively in MPPRS to make the complexity of the task allocation reduction and achieve the solving efficiency improvement. In MPPRS, a production planning result can be generated by the production planning layer at the beginning of every period T according to the static manufacturing information. The real-time scheduling result can be generated by the real-time scheduling layer according to the real-time manufacturing data of the workshop at each time t.

5.1 Infinitely repeated game model

The studied MPPRS problem can be regarded as an infinitely repeated game in which the same stage-game repeats periodically ($t_g = 0,1,2,\cdots$) when the number of repetitions is infinite. A stage-game is a tuple $\{N, A, \varphi\}$, where N is the set of players. Player $i \in N$ has a finite set A_i of actions. Each player i choose a certain action $a_i \in A_i$; the resulting vector form an action profile. The action profile is then executed, and the corresponding stage-game outcome is realized. To build the infinitely repeated game model, firstly, it is necessary to determine three elements in stage-game: players, strategies, and payoffs. The detailed descriptions are as follows:

$$G = \{N, A, \varphi\}$$

$N = \{1,\cdots,n\}$ is the set of players. In this problem, two objectives of production planning layer and three objectives of real-time scheduling layer correspond to players respectively.

$A = \prod_{i \in N} A_i$ and A_i is the set of player i's strategies. In production planning layer, the operations of the TP to the strategies are denoted as a strategy profile. In real-time scheduling layer, the operations of the RSTP are the strategies.

$\varphi = (\varphi_1, \cdots, \varphi_n)$ and φ_i is the set of stage-game payoff function for player i. In this problem, whether production planning layer or real-time scheduling layer, the stage-game payoff function of each player is the reciprocal value of the corresponding objective function.

For any stage-game $G = \{N, A, \varphi\}$, the infinitely repeated game is denoted by G^{∞}. In G^{∞}, at every stage ($t_g = 0,1,2,\cdots$), every player must decide his own action $a_i^{t_g} \in A_i$. Due to the number of repetitions of the stage-game is unknown, the weighted sum is used to calculate the total payoff function of each player of the infinitely repeated game (Leyton-Brown and Shoham, 2008):

$$U_i(s) = (1-\delta)\sum_{t_g=0}^{\infty} \delta^{t_g} \varphi_i \tag{10}$$

Where $U_i(s)$ is the total payoff, δ is the weighting factor, which only accepts values between 0 and 1.

5.2 Production planning layer

Production planning layer can generate a production planning at the beginning of every period T in the static manufacturing environment by using the infinitely repeated game. The input of this layer includes the task information and the real-time scheduling result of the last period. At the beginning of T_h, the implementation steps of the infinitely repeated game based production planning are shown in Fig.2. The detailed steps are described as follows.

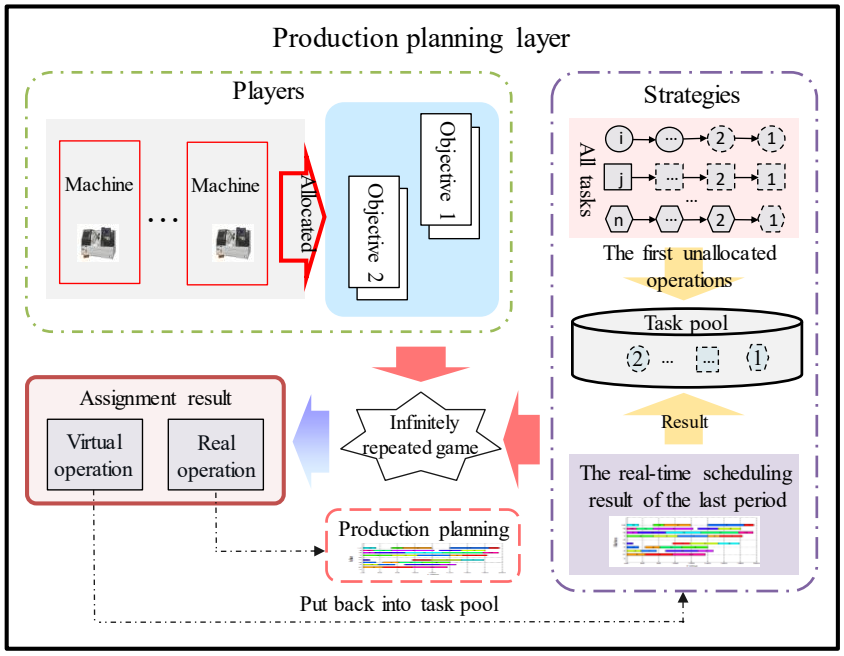

Fig.2. The implementing processes of the production planning layer

Step 1: The machines of the workshop are assigned to the two objectives of production planning layer in turn until the assignment ends.

Step 2: The two objectives work as two players and make their own decisions independently.

Step 3: Put the first unallocated operations of all tasks into the TP based on the real-time scheduling result of the last period. The operations in the TP called the strategies of the players. Thus, the operations of the TP can be competed by each machine.

Step 4: According to Eqs. (2)-(4) calculate the stage-game payoff of each player from various possible strategies combinations.

Step 5: Get the infinitely repeated game solution according to a grim trigger strategy and a folk theorem, which are detailed in Section 5.4.

Step 6: According to the result of Step 5, some or all operations are assigned to the machines. In order to ensure that the allocation of the operation is optimal, only one operation is assigned to the corresponding machine at a time. This operation is called real operation and others are called virtual operations, which are put back into the TP. This step distinguishes between real operation and the virtual operation according to the following rule in Fig.3, which refers to the classic scheduling allocation rules (e.g. Shortest Processing Time, Most Operation Number Remaining, Random).

//The steps of rules
Input: The assigned result of the operations.
Start
For each O_{ij} which is assigned to each M_k
If $t_{ijkc}+t_{ijkt}<t_{ijlc}+t_{ijlt}$ $(l=1,2,k-1,k+1,\cdots,m)$
{Put the O_{ij} into operation pool 1
If the number of operations in operation pool 1 is greater than 1
{Select the O_{ij} of the earliest completion time which belong to operation pool 1, and switch to operation pool 2
If the number of operations in operation pool 2 is still greater than 1
{Select the O_{ij} of the most operation number remaining which belong to operation pool 2, and switch to operation pool 3
If the number of operations in operation pool 3 is still greater than one
{Select one O_{ij} according to the random rule and this O_{ij} is put into PPQ of M_k
Else O_{ij} which belongs to operation pool 3 is put into PPQ of M_k.
}
Else O_{ij} which belongs to operation pool 2 is put into PPQ of M_k
}
Else O_{ij} which belongs to operation pool 1 is put into PPQ of M_k
}
Else select others infinitely repeated game solution
End
Output: Only one operation is put into PPQ of machines in the end.

Fig.3. Allocated rules

Repeat the above steps until all operations are put into the corresponding PPQ of the machine. The output of this layer is the task PPQ. This output determines which operation should be finished in which real-time scheduling period.

5.3 Real-time scheduling layer

In the production planning layer, the PPQ of the machine can be known at the beginning of T_h, meaning that the real-time scheduling layer knows which operations must be processed within T_h. Thus, it is used to assign operations from production planning layer to the most optimal machine. The inputs of this layer include the PPQ of each machine decided from production planning layer as well as the various exceptional events and machine status. This section describes the infinitely repeated game based process of the operations assignment in the real-world workshop environment at each time t ($t \in T_h$). In addition, before solving real-time scheduling of T_h, pick out all operations which should be processed within T_h according to the PPQ of each machine and put into the TP1. Thus, at each time t, the detailed steps of the real-time scheduling are introduced as follows and shown in Fig.4.

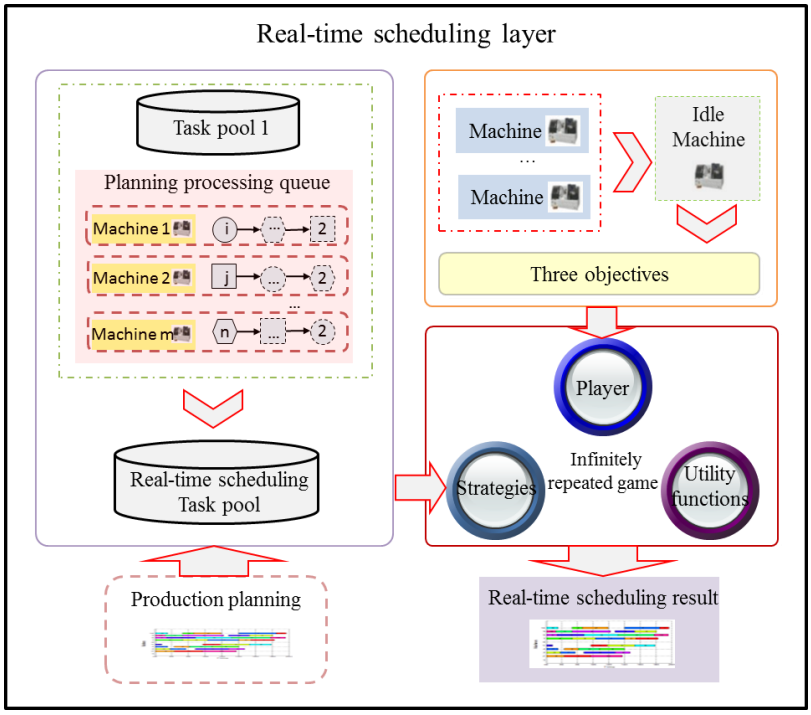

Fig.4. The implementing processes of the real-time scheduling layer

Step 1: Pick out the M_{idle}^{t} from M, then M_{idle}^{t} are allocated to the f_1^r, f_2^r, and f_3^r in turn until the assignment completes.

Step 2: The f_1^r, f_2^r, and f_3^r are seen as three players respectively. They can make their own decisions.

Step 3: Pick out the O_{option}^{t} from TP1 and put it into the RSTP. The O_{option}^{t} satisfy the following constraints:

(1) Operations precedence constraints of tasks;

(2) PPQ precedence constraints of production planning on each M_k;

In addition, there is a special case as follows: if an operation is assigned to M_k in production planning layer and this operation is not assigned to M_k in real-time scheduling layer in the end, the

next operation which meets the PPQ precedence constraints can be known as O^t_{option} , even if the previous operation does not complete in real-time scheduling layer.

The O^t_{option} in the RSTP are known as the strategies. Thus, M^t_{idle} which is assigned to the f^r_i (i=1, 2, 3) can choose the most optimal operations from the RSTP.

Step 4: According to Eqs. (7)-(9) calculate the payoff of each player from each possible combination of strategies.

Step5: Find the infinitely repeated game solutions, which are detailed in Section 5.4. Then, the O^t_{option} are put into the process queue of most optimal machines.

Step 6: Repeat step 1 to step 5 until the end of T_h .

The output of this layer is the task process sequence according to the real-time manufacturing data. Each machine should follow this output to process operation. When exceptional events occur, the above method also can reduce the effect of the exceptions in time.

5.4 Infinitely repeated game solution

In the dynamic game, the sub-game perfect Nash equilibrium (SPNE) is considered to be a balance point where there is no reason for each player to change his behaviour. Many dynamic games use the backward induction method to find the SPNE. However, the infinitely repeated game does not exist in the last stage of a game, and the backward induction method no longer works. Thus, in this study, the grim trigger strategy is introduced.

The grim trigger strategy is defined as follows: in the grim trigger strategy, a^* refers Nash equilibrium (NE) of the stage-game, and $a' = (a'_1, a'_2, \cdots, a'_n)$ is a strategy combination of G. For any player i with $\varphi_i(a') > \varphi_i(a^*)$, the player i that can adopt the strategy is described as follows: (1) a'_i is chosen in the first stage-game, and this strategy is also chosen in all subsequent stages-game; (2) If any other player chooses the maximum payoff of the period instead of a'_j before the t stage-game, player i will select the a^*_i and repeat until the end.

From the above analysis, it can be seen that the SPNE of infinite repeated game consists of two parts: (1) the NE or SPNE of the stage game, namely a^* in the grim trigger strategy, and this equilibrium solution is called the non-cooperative equilibrium solution; (2) namely a' in the combination of grim trigger strategy, and this equilibrium solution is called cooperative equilibrium solution. Since players can get more benefits under a' than under a^*, the SPNE of the infinitely repeated game required is a' in this paper.

In the grim trigger strategy, the δ of Eqs. (10) plays an essential role in the SPNE of the infinitely repeated game. If δ is bigger, the number of repetitions of the stage-game is larger and players are patient and cooperative. Thus, a specific SPNE of an infinitely repeated game, which could be called a cooperative equilibrium solution, can be obtained as soon as the δ is large

enough.

Friedman (1971) proposed a cooperative equilibrium solution existence theorem (a folk theorem) as follows:

Let a^* be an equilibrium (an equilibrium of the stage game) with payoffs $\varphi^* = (\varphi_1^*, \varphi_2^*, \cdots, \varphi_n^*)$. Then for any $v = (v_1, v_2, \cdots, v_n) \in V$ (V is the set of feasible payoffs of the stage-game) with $v_i > \varphi_i^*$ of all player i, there is a $\delta \in (0,1)$ that there is an SPNE of G^∞ with payoffs v.

Thus, the payoffs of SPNE can be gained quickly to find the appropriate δ, which means that the SPNE can be known. In this study, a folk theorem-based solution procedure is proposed. The solution algorithm is summarized in Fig. 5.

//Algorithm for infinitely repeated game equilibrium solution
Input: A static stage-game procedure.
Start
Step 1. Determine an Nash equilibrium (NE) solution of G according to each player's payoff. The contents of NE solution are described in our previous paper (Zhang et al., 2017a). The NE solution denotes as a^* and the corresponding payoffs are $\varphi^* = (\varphi_1^*, \varphi_2^*, \cdots, \varphi_n^*)$;
Step 2. Find out $v = (v_1, v_2, \cdots, v_n) \in V$ from φ with $v_i > \varphi_i^*$ for all player i;
Step 3. Find out the corresponding strategies a_i' of v_i;
Step 4. Suppose the triggering strategy is adopted in the first time of infinitely repeated game and calculate the δ according to $(1-\delta)\sum_{t_g=0}^{\infty} \delta^{t_g} v_i > \max g_i + \varphi_i^*(\delta + \delta^2 + \cdots)$.
END
Output: A sub-gameperfect equilibrium (a') of G^∞ and δ.

Fig.5. Solution algorithm

6. Case study

This section describes the MPPRS method through a simulation example to demonstrate the effectiveness and performance of the proposed method. All the simulation examples were performed with MATLAB 2010b on a personal computer with Intel Core i5, 3.1GHz CPU and 4GB RAM. The real-time data captured in the IoMT environment were put into the MySQL database that is a relational database management system. The stored data in this database can be retrieved during real-time scheduling.

The basic data of simulation example come from Kacem's instance (Kacem et al., 2002) and the dimensions of the instance are 8 tasks × 8 machines. In order to obtain the total energy consumption of production during the manufacturing execution stage, it's different from the Kacem's instance that the cutting power, the tool change time and the cutting time are presented in this study. Here, the sum of the tool change time and the cutting time is processing time. In additional, J_9 in this simulation is considered as a rush order. The detailed data of the simulation example is shown in Table 2. In Table 2, the three figures (i.e. a/b/c) in O_{ij} row and M_k column represent that the tool change time, the cutting time and the cutting power respectively, when the operation 'j' of task 'i' is processed on machine 'k'. Table 3 shows the machine idle power, which

comes from He et al. (2015).

Table 2 The instance of MPPRS

Tasks	Operations	Available machine (tool change time [hour]/ cutting time [hour]/ cutting power [KW])							
		M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
J_1	O_{11}	1/4/1.3	1/2/1.8	1/4/3.2	2/1/1.1	1/2/1.1	-	4/6/0.8	2/7/1.1
	O_{12}	2/8/1.3	-	2/3/3.4	3/5/3.2	1/2/0.8	3/6/0.8	4/5/0.9	2/4/1.3
	O_{13}	-	3/7/1.8	-	2/3/1.4	4/2/0.7	1/1/0.9	2/2/1.2	3/2/1.3
J_2	O_{21}	2/3/1.6	4/3/2.1	2/1/2.6	2/7/1.5	1/7/1.2	-	5/4/1.1	-
	O_{22}	-	3/5/2.4	1/4/2.4	1/1/1.6	2/4/1.4	2/5/1.2	2/8/1.3	1/8/1.4
	O_{23}	-	4/6/2.3	-	2/3/1.5	5/1/0.9	2/2/1.8	0.5/0.5/1.4	3/4/1.3
	O_{24}	3/7/1.4	3/5/1.8	2/7/2.4	4/2/3.2	1/3/0.8	3/4/1.7	-	-
J_3	O_{31}	3/7/2.1	-	-	1/6/1.5	3/3/0.7	4/1/1.6	1/1/1.3	3/1/1.2
	O_{32}	-	2/8/1.9	1/5/2.6	3/1/1.6	2/6/1.2	5/4/1.7	2/8/1.4	-
	O_{33}	0.5/0.5 /1.4	1/3/2.5	2/3/4.2	2/4/1.4	-	2/8/1.3	-	2/5/0.8
J_4	O_{41}	1/2/1.3	0.2/0.8/2.4	2/4/3.2	1/4/2.1	2/7/1.3	3/4/1.7	5/3/1.3	2/2/1.1
	O_{42}	3/9/1.4	4/7/2.6	3/4/4.2	5/3/3.2	2/8/1.5	3/2/0.8	5/1/1.2	3/6/1.3
	O_{43}	2/2/1.4	4/2/3.7	1/1/3.2	3/7/1.5	2/1/0.8	2/7/0.7	4/1/1.4	2/5/1.8
J_5	O_{51}	1/2/1.3	2/4/1.2	3/4/2.4	4/4/1.2	4/5/0.8	-	4/6/1.3	-
	O_{52}	2/8/1.2	-	3/4/2.8	3/1/2.1	2/7/1.3	5/3/0.7	4/2/1.3	-
	O_{53}	-	3/6/3.2	2/6/3.2	2/5/1.8	1/3/1.2	1/1/1.2	2/5/1.4	-
J_6	O_{54}	2/9/2.1	2/6/1.6	-	2/4/1.7	4/3/1.5	2/3/1.3	2/1/1.3	2/4/1.3
	O_{61}	1/5/1.4	2/5/1.7	0.3/0.7/4.2	1/3/1.6	2/4/0.8	2/7/1.4	-	3/7/1.3
	O_{62}	4/7/1.3	-	3/6/3.2	2/7/1.4	5/4/0.9	2/5/0.9	1/5/1.3	2/2/1.3
	O_{63}	5/5/1.4	2/3/2.1	4/5/2.4	3/7/1.5	2/9/1.2	-	2/8/1.2	-
J_7	O_{71}	2/3/1.1	2/2/2.2	1/1/3.2	2/4/1.3	4/3/1.3	-	2/8/0.8	-
	O_{72}	-	4/5/2.5	-	5/4/1.4	6/5/0.8	2/7/1.6	2/8/1.3	2/3/1.4
	O_{73}	-	2/6/2.4	4/5/4.2	2/1/1.2	3/5/1.2	2/4/2.1	-	5/5/1.6
J_8	O_{81}	1/1/1.4	1/7/3.2	2/3/2.2	2/7/1.4	-	1/3/1.2	-	3/7/1.8
	O_{82}	2/5/1.3	2/2/1.7	3/4/2.9	4/4/1.4	4/5/1.1	-	2/8/1.3	-
	O_{83}	2/7/1.4	2/7/3.2	-	6/2/1.2	3/2/0.8	5/1/1.3	2/5/1.4	0.5/0.5/1.3
	O_{84}	4/5/1.7	-	1/2/4.1	4/3/1.2	0.2/0.8/0.9	2/3/1.4	2/6/0.9	-
J_9	O_{91}	2/3/1.3	2/5/2.4	4/4/3.2	3/2/1.1	4/1/1.1	-	3/4/1.6	2/2/1.2
	O_{92}	2/2/1.6	2/5/2.3	4/10/2	2/2/1.2	1/3/1.3	2/4/1.1	-	3/7/1.3
	O_{93}	1/4/1.2	1/3/2.2	4/2/3.2	3/8/1.3	2/5/2.1	3/10/1.3	1/4/1.3	3/2/1.5

Table 3 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, the implement processes of MPPRS consist of five main steps, which are shown as follows:

(1) The equation (1) is used to calculate the time unit of T. Here, the n' is 3.

(2) At the beginning of T_h, all remaining unprocessed operations are put into the PPQ of the machines. Thus, production planning is generated based on the static environment for the real-time scheduling within T_h. For each machine, operations are assigned to the PPQ of the machine using the method of production planning layer (refer to Section 5.2). Then, the result of production planning is released to the real-time scheduling layer.

(3) After putting all remaining unprocessed operations into the PPQ of the machine, a real-time scheduling method (refer to Section 5.3), is used to assigned operations to the most appropriate machine in the real-time manufacturing workshop at each time t ($t \in T_h$).

(4) At the beginning of T_{h+1}, the production planning layer obtains the unprocessed operations

and the real-time scheduling result within T_h. Then, new production planning can be generated according to the above information. After that repeat the above process until finish all operations. Table 4 shows the real-time scheduling process in a static environment.

(5) During the processing stage, real-time manufacturing data can be known by each manufacturing resource. If the exceptions occur, the corresponding method can reduce the effect of exceptions event in time, which is presented in Section 5.

Table 4. The real-time scheduling process in a static environment

Time	Result
0	$O_{81}\rightarrow M_1$; $O_{41}\rightarrow M_2$; $O_{71}\rightarrow M_3$; $O_{11}\rightarrow M_4$; $O_{31}\rightarrow M_7$;
1	$O_{42}\rightarrow M_6$
2	$O_{51}\rightarrow M_1$; $O_{82}\rightarrow M_2$; $O_{61}\rightarrow M_3$
3	$O_{21}\rightarrow M_3$; $O_{32}\rightarrow M_4$; $O_{12}\rightarrow M_5$; $O_{62}\rightarrow M_8$
4	None
5	None
6	$O_{43}\rightarrow M_3$; $O_{13}\rightarrow M_6$
7	$O_{33}\rightarrow M_1$; $O_{63}\rightarrow M_2$; $O_{22}\rightarrow M_4$; $O_{72}\rightarrow M_8$
8	None
9	$O_{52}\rightarrow M_4$; $O_{23}\rightarrow M_7$
10	$O_{24}\rightarrow M_5$
11	None
12	$O_{83}\rightarrow M_8$
13	$O_{73}\rightarrow M_4$; $O_{53}\rightarrow M_6$
14	$O_{84}\rightarrow M_5$
15	$O_{54}\rightarrow M_7$

To demonstrate the effectiveness of the MPPRS for the LFSP, we compare the MPPRS with the existing scheduling methods including AL+CGA (Kacem et al., 2002b), PSO+TS (Zhang et al., 2009), AIA (Bagheri et al., 2010), and P-DABC (Li et al., 2011) in a static workshop environment. The AL+CGA refers to a genetic algorithm controlled by the assigned model which was generated by the approach of localization (AL) to multi-objective flexible job shop. The PSO+TS uses the PSO algorithm to assign operations on machines and to schedule operations on each machine, and TS is applied to local search for the scheduling sub-problem originating from each obtained solution. The AIA uses several strategies for generating the initial population and selecting the individuals for reproduction. Different mutation operators are also utilized for reproducing new individuals. In the P-DABC, a well-designed crossover operator is presented for information sharing among the employed bees. An external Pareto archive set is developed to memorize the nondominated solutions found so far. Although the above methods optimize multiple objectives of a flexible job shop, only two objectives are considered in the instance of *kacem* (8 tasks × 8 machines), such as the makespan and the total workload of machines. The total energy consumption of production is not considered as an objective. To compare with our proposed method, we calculate the values of the total energy consumption of production base on the existing optimized result of each scheduling method. The results of the comparison between MPPRS and existing scheduling methods are shown in Table 5.

Table 5. Results of the comparison between MPPRS and existing scheduling methods

Objectives	AL+CGA	AIA	PSO+TS	P-DABC	Proposed approach

f_1^r [hour]	15	14	14	14	18
f_2^r [hour]	79	77	77	77	75
f_3^r [kWh]	121.59	118.01	117.67	119.30	112.91

As seen in Table 5, the proposed approach in this study obtained the best f_2^r, which is 75 hours. However, the best solution and the worst solution are 77 hours and 79 hours respectively in existing scheduling methods. The minimum improvement is 2.6%, and the maximum improvement is 5.1%. According to Table 5, compared with the existing scheduling method, f_1^r in our proposed method is slightly increased. However, the f_3^r obtained by our proposed method is 112.91 kWh, which means that the minimum improvement is 4.0%, and the maximum improvement is 7.1% compared to the other four algorithms. It can be seen that although less makespan can reduce the total idle time of machines, it may increase the cutting energy consumption resulting in larger total energy consumption of production. Thus, it can be seen from the above analysis that the proposed method is effective to realize the tri-objective optimization to minimize the makespan, the total workload of machines and the total energy consumption in the LFJS without considering the exceptional events.

In order to illustrate the performance of the proposed method for the real-time LFJS scheduling under the dynamic manufacturing environment, it is compared with several traditional dynamic scheduling methods, such as an NSGA-II+right shift rescheduling method (W. Wang et al., 2018), and completely reactive scheduling methods, which used a machine assignment rule to assign operations to machines and the machine chooses the operations according to a heuristic dispatching rule. In this study, four common heuristic dispatching rules which are the shortest processing time (SPT), the longest processing time (LPT), the most work remaining (MWKR) and the least work remaining (LWKR) are employed. In addition, three machine assignment rules (MAR) are considered: (1) each operation is assigned to the available machines with the shortest processing time, (2) each operation is assigned to its available machines with minimum workload currently, and (3) each operation is assigned to the random alternative machine. These three machine assignment rules can be named as MAR1, MAR2, and MAR3.

In order to compare the simulation results of each scheduling method, three test cases are proposed and the results are shown in Table 6-8: (1) Test case 1: M_2 and M_5 break down at time t_1=3 and t_2=5. The recovered time are t_3=6 and t_4=7 respectively. Test case 2: J_9 are added at time t_5=8. Test case 3: the above three exceptions are involved at the same time.

Table 6. Test case 1

Scheduling methods	f_1^r [hour]	f_2^r [hour]	f_3^r [kWh]
MAR1+SPT	19	73	113.91
MAR1+LPT	22	73	114.98
MAR1+MWKR	21	73	119.32
MAR1+LWKR	22	73	128.89

MAR2+SPT	26	107	173.56
MAR2+LPT	31	104	223.11
MAR2+MWKR	29	112	236.89
MAR2+LWKR	32	124	186.54
MAR3+SPT	23	91	156.98
MAR3+LPT	25	93	176.54
MAR3+MWKR	28	98	203.18
MAR3+LWKR	31	106	197.26
NSGA-II+right shift rescheduling	25	79	153.26
Proposed method	18	75	113.21

Table 7. Test case 2

Scheduling methods	f_1^r [hour]	f_2^r [hour]	f_3^r [kWh]
MAR1+SPT	21	85	119.76
MAR1+LPT	28	85	135.11
MAR1+MWKR	22	85	129.84
MAR1+LWKR	24	85	132.30
MAR2+SPT	29	132	230.56
MAR2+LPT	32	143	328.69
MAR2+MWKR	33	156	276.34
MAR2+LWKR	37	149	257.83
MAR3+SPT	27	113	178.56
MAR3+LPT	30	124	239.23
MAR3+MWKR	29	142	214.91
MAR3+LWKR	35	137	221.48
NSGA-II+right shift rescheduling	27	105	157.53
Proposed method	19	87	117.53

Table 8. Test case 3

Scheduling methods	f_1^r [hour]	f_2^r [hour]	f_3^r [kWh]
MAR1+SPT	20	85	124.36
MAR1+LPT	28	85	133.98
MAR1+MWKR	21	85	125.86
MAR1+LWKR	23	85	132.78
MAR2+SPT	27	121	212.36
MAR2+LPT	30	130	229.56
MAR2+MWKR	30	145	218.29
MAR2+LWKR	34	137	236.59
MAR3+SPT	24	101	163.38
MAR3+LPT	27	114	223.12
MAR3+MWKR	27	133	203.82
MAR3+LWKR	31	130	206.34
NSGA-II+right shift rescheduling	28	112	180.24
Proposed method	19	87	122.41

The results of test case 1 are shown in Table 6. By comparing the solutions, it can be seen that the proposed method is better than the traditional dynamic scheduling methods. For example, the f_1^r value is 18 hours in our proposed method and the maximum improvement is 43.8% compared with the traditional scheduling methods. Although the proposed method is not dominant to the optimization of f_2^r compared with the completely reactive scheduling method with the MAR1, the maximum improvement is 55.36%, and the minimum improvement is 6.25% compared with other traditional dynamic scheduling methods. In addition, the proposed method can reduce f_3^r obtained by traditional dynamic scheduling method from the maximum 236.89kWh and the minimum

113.91kWh to 113.21kwh (-52.21% and -0.61%).

The results of test case 2 are shown in Table 7. These results further demonstrate that the proposed method obtains better solutions than the traditional dynamic scheduling methods. From Table 7, it is clear that the f_1^r obtained by our proposed method is 19 hours. However, the minimum f_1^r obtained by the traditional dynamic scheduling methods is 21 hours. Compared with the completely reactive scheduling method with the MAR1, f_2^r in our proposed method is slightly increased. By the other existing dynamic scheduling methods, the best f_2^r is 105 hours and the worst f_2^r is 156 hours. The maximum improvement is 44.23%. As the same to the test case 1, the f_3^r also can be decreased.

Test case 3 can be viewed as a comprehensive form of test case 1 and test case 2, where contain both types of exceptions. The results are given in Table 8. It can be seen that the proposed method is not dominant to the optimization of f_2^r compared with the completely reactive scheduling method with the MAR1, however, the proposed method has better solutions than other traditional dynamic scheduling methods. In addition, compared with the traditional dynamic scheduling methods, it can be found that f_1^r and f_3^r can achieve the maximum improvement is 44.1%, and 48.3%, and the minimum improvement is 5%, and 1.6% respectively.

It can be seen from the above three test cases that the completely reactive scheduling method with the MAR1 can obtain better f_2^r than the proposed method. The reason is that the MAR1 always assigns an operation to its alternative machine with the minimum processing time, which tends to reduce the total workload of machines. But on the other hand, it may lead to long waiting queues in specific machines, which results in a long finishing time of all the operations.

In order to further evaluate the performance of the proposed method, a generic simulation is set up where for each of 100 runs, k ($1 \le k \le 8$) random machines are chosen to break at a random time within the production horizon and (or) one rush order are inserted randomly. The machine repair time is assumed to follow a random integer uniformly distributed on [1, 4]. We compare the average values of objectives found by the proposed method and the traditional dynamic scheduling methods. The experimental results were obtained by all methods in Table 9.

Table 9. The average values of objectives

Scheduling methods	f_1^r [hour]	f_2^r [hour]	f_3^r [kWh]
MAR1+SPT	42	107	329.35
MAR1+LPT	49	107	348.04
MAR1+MWKR	43	107	335.02
MAR1+LWKR	46	107	351.34
MAR2+SPT	49	148	315.50
MAR2+LPT	54	167	380.47
MAR2+MWKR	53	166	353.85
MAR2+LWKR	57	166	346.04
MAR3+SPT	47	160	376.32
MAR3+LPT	50	139	332.98

MAR3+MWKR	50	152	317.31
MAR3+LWKR	55	153	328.38
NSGA-II+right shift rescheduling	49	127	273.69
Proposed method	39	110	227.74

Table 9 reveals that although the proposed method in terms of the total workload of machines is not always better than the traditional dynamic scheduling method, the proposed method in terms of the makespan and the total energy consumption of production can obtain better results. This further verifies our proposed method that real-time manufacturing information based real-time scheduling is more likely to generate satisfactory results.

It is worth mentioning that the CPU time plays a critical role in prompt effective real-time scheduling. Table 10 lists the mean CPU time of all methods at each time t or rescheduling point. It can be seen that the mean CPU time of some completely reactive scheduling methods is much shorter than the proposed method. However, the proposed method can improve the schedule efficiency and reduce energy consumption at the same time. Meanwhile, with acceptable CPU time (1-2min), the CPU time of the proposed method is acceptable for the real production system.

Table 10. CPU time comparisons of all scheduling methods

Scheduling methods	Mean CPU time (s)
MAR1+SPT	2.352
MAR1+LPT	5.321
MAR1+MWKR	3.562
MAR1+LWKR	4.234
MAR2+SPT	3.986
MAR2+LPT	10.351
MAR2+MWKR	4.136
MAR2+LWKR	10.417
MAR3+SPT	3.536
MAR3+LPT	11.248
MAR3+MWKR	4.196
MAR3+LWKR	10.348
NSGA-II+right shift rescheduling	38.567
Proposed method	4.146

Based on the above analysis, compared with the traditional dynamic scheduling methods, the proposed method considers the optimization objective of production planning before real-time scheduling and conducts real-time scheduling at each time t according to the production planning. It not only considers the needs of workshop managers but also significantly reduces the complexity of scheduling calculation in dynamic scheduling.

In addition, compared with the traditional dynamic scheduling method, at any time, each machine can obtain the optimal operation. Therefore, the complexity of real-time scheduling is stable with the increasing of operation because only one optimal operation is selected for one machine at each time by using infinite repeated games. Thus, the proposed method is better than the traditional dynamic scheduling methods in a real manufacturing environment when exceptions occur.

7. Conclusions and future work

In this study, an MPPRS method is put forward on the basis of the IoT technology for LFJS.

Three contributions are important. Firstly, a real-time production scheduling method is proposed based on real-time manufacturing data. all machines of the workshop can send their real-time data and request the operations of the RSTP. At the same time, operations constantly interact with the machines. Then, the machines can acquire corresponding operations. Secondly, the MPPRS method is developed. It includes two parts, namely the production planning layer and real-time scheduling layer. In the production planning layer, a new schedule is generated as a production planning at the beginning of every period T in the static manufacturing environment. In the real-time scheduling layer, each operation is assigned to the most appropriate machine based on the real-time manufacturing data of the workshop. The third contribution is a proposal of the infinitely repeated game optimization method for MPPRS. Under the general game condition, the strategy with the greatest overall payoff may not be a stable and usable one. This study proposes the infinitely repeated game optimization model. It can use the interaction between short-term interests and long-term incentives and promote the cooperation between players through the reputation mechanism in the game. Therefore, a better optimal result can be obtained compared with the general game. The case study shows that the performance of the proposed method in terms of the production efficiency and energy-saving are effective and efficient compared against the traditional dynamic scheduling methods.

Future research will focus on the real-time data based production scheduling problem at the enterprise level. In addition, the integration of this study with the real-time data-driven material delivery will be also considered.

Acknowledgements

Authors would like to acknowledge the financial support of the National Key Research and Development Program of China (2018YFB1703402), the Seed Foundation of Innovation and Creation for Graduate Students in Northwestern Polytechnical University (ZZ2019011).

References

Agnetis, A., Nicosia, G., Pacifici, A., Pferschy, U., 2015. Scheduling two agent task chains with a central selection mechanism. J. Sched. 18, 243–261. https://doi.org/10.1007/s10951-014-0414-9

Arasteh, A., Naini, S.G.J., Aliahmadi, A., 2014. Considering the game-theoretic approach and ultra combinative costs on scheduling. Int. J. Adv. Manuf. Technol. 70, 1473–1485. https://doi.org/10.1007/s00170-013-5388-9

Araz, Ö.U., Salum, L., 2010. A multi-criteria adaptive control scheme based on neural networks and fuzzy inference for DRC manufacturing systems. Int. J. Prod. Res. 48, 251–270. https://doi.org/10.1080/00207540802471256

Aytug, H., Lawley, M.A., McKay, K., Mohan, S., Uzsoy, R., 2005. Executing production schedules in the face of uncertainties: A review and some future directions, in: European Journal of Operational Research. pp. 86–110. https://doi.org/10.1016/j.ejor.2003.08.027

Babayan, A., He, D., 2004. Solving the n -job 3-stage flexible flowshop scheduling problem using an agent-based approach. Int. J. Prod. Res. 42, 777–799. https://doi.org/10.1080/00207540310001602946

Bagheri, A., Zandieh, M., Mahdavi, I., Yazdani, M., 2010. An artificial immune algorithm for the flexible job-shop scheduling problem. Futur. Gener. Comput.

Calleja, P., Est??vez-Fern??ndez, A., Borm, P., Hamers, H., 2006. Job scheduling, cooperation, and control. Oper. Res. Lett. 34, 22–28. https://doi.org/10.1016/j.orl.2005.01.007

Cassettari, L., Bendato, I., Mosca, M., Mosca, R., 2017. Energy Resources Intelligent Management using on line real-time simulation: A decision support tool for sustainable manufacturing. Appl. Energy 190, 841–851. https://doi.org/10.1016/j.apenergy.2017.01.009

Chen, Y.X., 2016. INTEGRATED OPTIMIZATION MODEL FOR PRODUCTION PLANNING AND SCHEDULING WITH LOGISTICS CONSTRAINTS. Int j simul Model 15, 711–720. https://doi.org/10.2507/IJSIMM15(4)CO16

Church, L.K., Uzsoy, R., 1992. Analysis of periodic and event-driven rescheduling policies in dynamic shops. Int. J. Comput. Integr. Manuf. 5, 153–163. https://doi.org/10.1080/09511929208944524

Dai, M., Tang, D., Xu, Y., Li, W., 2015. Energy-aware integrated process planning and scheduling for job shops. Proc. Inst. Mech. Eng. Part B J. Eng. Manuf. 229, 13–26. https://doi.org/10.1177/0954405414553069

Davis, W.J., Thompson, S.D., 1993. Production planning and control hierarchy using a generic controller. IIE Trans. (Institute Ind. Eng. 25, 26–45. https://doi.org/10.1080/07408179308964302

Friedman, J.W., 1971. A Non-cooperative Equilibrium for Supergames. Rev. Econ. Stud. 38, 1. https://doi.org/10.2307/2296617

Guo, L., Wang, S., Kang, L., Cao, Y., 2015. Agent-based manufacturing service discovery method for cloud manufacturing. Int. J. Adv. Manuf. Technol. 81, 2167–2181. https://doi.org/10.1007/s00170-015-7221-0

Han, Z., Zhu, Y., Ma, X., Chen, Z., 2016. Multiple rules with game theoretic analysis for flexible flow shop scheduling problem with component altering times. Int. J. Model. Identif. Control 26, 1. https://doi.org/10.1504/IJMIC.2016.077749

He, Y., Li, Y., Wu, T., Sutherland, J.W., 2015. An energy-responsive optimization method for machine tool selection and operation sequence in flexible machining job shops. J. Clean. Prod. 87, 245–254. https://doi.org/10.1016/j.jclepro.2014.10.006

Holloway, C.A., Nelson, R.T., 1974. Job Shop Scheduling with Due Dates and Overtime Capability. Manage. Sci. 21, 68–78. https://doi.org/doi:10.1287/mnsc.20.9.1264

Hondo, H., 2005. Life cycle GHG emission analysis of power generation systems: Japanese case, in: Energy. https://doi.org/10.1016/j.energy.2004.07.020

Hosseinabadi, A.A.R., Siar, H., Shamshirband, S., Shojafar, M., Nasir, M.H.N.M., 2015. Using the gravitational emulation local search algorithm to solve the multi-objective flexible dynamic job shop scheduling problem in Small and Medium Enterprises. Ann. Oper. Res. 229, 451–474. https://doi.org/10.1007/s10479-014-1770-8

Hu, Y., Rao, S.S., 2009. Game-theory approach for multi-objective optimal design of stationary flat-plate solar collectors. Eng. Optim. 41, 1017–1035. https://doi.org/10.1080/03052150902890064

Huang, B., Wang, W., Ren, S., Zhong, R.Y., Jiang, J., 2019. A proactive task dispatching method based on future bottleneck prediction for the smart factory. Int. J. Comput. Integr. Manuf. https://doi.org/10.1080/0951192X.2019.1571241

Jula, P., Kones, I., 2013. Continuous-time algorithms for scheduling a single machine with sequence-dependent setup times and time window constraints in coordinated chains. Int. J. Prod. Res. 51, 3654–3670. https://doi.org/10.1080/00207543.2012.757666

Kacem, I., Hammadi, S., Borne, P., 2002a. Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 32, 1–13. https://doi.org/10.1109/TSMCC.2002.1009117

Kacem, I., Hammadi, S., Borne, P., 2002b. Pareto-optimality approach for flexible job-shop scheduling problems: Hybridization of evolutionary algorithms and fuzzy logic. Math. Comput. Simul. 60, 245–276. https://doi.org/10.1016/S0378-4754(02)00019-8

Kouiss K, Pierreval H, M.N., 1997. Using multi-agent architecture in FMS for dynamic scheduling. J. Intell. Manuf. 8, 41–47. https://doi.org/{10.1023/A:1018540317470

Kundakci, N., Kulak, O., 2016. Hybrid genetic algorithms for minimizing makespan in dynamic job shop scheduling problem. Comput. Ind. Eng. 96, 31–51. https://doi.org/10.1016/j.cie.2016.03.011

Lei, D., 2008. A Pareto archive particle swarm optimization for multi-objective job shop scheduling. Comput. Ind. Eng. 54, 960–971. https://doi.org/10.1016/j.cie.2007.11.007

Lei, D., Zheng, Y., Guo, X., 2017. A shuffled frog-leaping algorithm for flexible job shop scheduling with the consideration of energy consumption. Int. J. Prod. Res. 55, 3126–3140. https://doi.org/10.1080/00207543.2016.1262082

Leyton-Brown, K., Shoham, Y., 2008. Essentials of Game Theory: A Concise Multidisciplinary Introduction. Synth. Lect. Artif. Intell. Mach. Learn. 2, 1–88. https://doi.org/10.2200/S00108ED1V01Y200802AIM003

Li, J.-Q., Pan, Q.-K., Gao, K.-Z., 2011. Pareto-based discrete artificial bee colony algorithm for multi-objective flexible job shop scheduling problems. Int. J. Adv. Manuf. Technol. 55, 1159–1169. https://doi.org/10.1007/s00170-010-3140-2

Li, X., Gao, L., Li, W., 2012. Application of game theory based hybrid algorithm for multi-objective integrated process planning and scheduling. Expert Syst. Appl. 39, 288–297. https://doi.org/10.1016/j.eswa.2011.07.019

Liu, B., Zhang, Y., Zhang, G., Zheng, P., 2019. Edge-cloud orchestration driven industrial smart product-service systems solution design based on CPS and IIoT. Adv. Eng. Informatics. https://doi.org/10.1016/j.aei.2019.100984

Lu, M.S., Romanowski, R., 2013. Multicontextual dispatching rules for job shops with dynamic job arrival. Int. J. Adv. Manuf. Technol. 67, 19–33. https://doi.org/10.1007/s00170-013-4765-8

Luh, P.B., Wang, J.H., Wang, J.L., Tomastik, R.N., Howes, T.D., 1997. Near-Optimal Scheduling of Manufacturing Systems with Presence of Batch Machines and Setup Requirements. CIRP Ann. - Manuf. Technol. 46, 397–402. https://doi.org/10.1016/S0007-8506(07)60851-8

Luo, H., Fang, J., Huang, G.Q., 2015. Real-time scheduling for hybrid flowshop in ubiquitous manufacturing environment, in: Computers and Industrial Engineering. Curran Associates Inc., pp. 12–23. https://doi.org/10.1016/j.cie.2014.09.019

Ma, S., Zhang, Y., Lv, J., Yang, H., Wu, J., 2019. Energy-cyber-physical system enabled management for energy-intensive manufacturing industries. J. Clean. Prod. 226, 892–903. https://doi.org/10.1016/j.jclepro.2019.04.134

Manupati, V.K., Deo, S., Cheikhrouhou, N., Tiwari, M.K., 2012. Optimal process plan selection in networked based manufacturing using game-theoretic approach. Int. J. Prod. Res. 50, 5239–5258. https://doi.org/10.1080/00207543.2012.682181

Meng, L., Zhang, C., Shao, X., Ren, Y., 2019. MILP models for energy-aware flexible job shop scheduling problem. J. Clean. Prod. 210, 710–723. https://doi.org/10.1016/j.jclepro.2018.11.021

Miyamoto, T., Noguchi, S., Yamashita, H., 2008. Selection of an optimal solution for multiobjective electromagnetic apparatus design based on game theory. IEEE Trans. Magn. 44, 1026–1029. https://doi.org/10.1109/TMAG.2007.915090

Mokhtari, H., Hasani, A., 2017. An energy-efficient multi-objective optimization for flexible job-shop scheduling problem. Comput. Chem. Eng. 104, 339–352. https://doi.org/10.1016/j.compchemeng.2017.05.004

Muhleman, A.P., Lockett, A.G., Farn, C.K., 1982. Job Shop Scheduling Heuristics and Frequncy of Scheduling. Int. J. Prduction Res. 20, 227–241.

Nguyen, S., Zhang, M., Johnston, M., Tan, K.C., 2015. Automatic programming via iterated local search for dynamic job shop scheduling. IEEE Trans. Cybern. 45, 1–14. https://doi.org/10.1109/TCYB.2014.2317488

Ning, T., Huang, M., Liang, X., Jin, H., 2016. A novel dynamic scheduling strategy for solving flexible job-shop problems. J. Ambient Intell. Humaniz. Comput. 7, 721–729. https://doi.org/10.1007/s12652-016-0370-7

Nouiri, M., Bekrar, A., Jemai, A., Niar, S., Ammari, A.C., 2015. An effective hybrid particle swarm optimization algorithm for flexible job-shop scheduling problem. J. Intell. Manuf. 1, 8–13. https://doi.org/10.1007/s10845-015-1039-3

Ouelhadj, D., Petrovic, S., 2009. A survey of dynamic scheduling in manufacturing systems. J. Sched. https://doi.org/10.1007/s10951-008-0090-8

Pierreval, H., Mebarki, N., 1997. Dynamic scheduling selection of dispatching rules for manufacturing system. Int. J. Prod. Res. 35, 1575–1591. https://doi.org/10.1080/002075497195137

Qian, C., Zhang, Y., Jiang, C., Pan, S., Rong, Y., 2020. A real-time data-driven collaborative mechanism in fixed-position assembly systems for smart manufacturing. Robot. Comput. Integr. Manuf. https://doi.org/10.1016/j.rcim.2019.101841

Qian, C., Zhang, Y., Liu, Y., Wang, Z., 2019. A cloud service platform integrating additive and subtractive manufacturing with high resource efficiency. J. Clean. Prod. https://doi.org/10.1016/j.jclepro.2019.118379

Raileanu, S., Anton, F., Iatan, A., Borangiu, T., Anton, S., Morariu, O., 2017. Resource scheduling based on energy consumption for sustainable manufacturing. J. Intell. Manuf. 28, 1519–1530. https://doi.org/10.1007/s10845-015-1142-5

Rangsaritratsamee, R., Ferrell, W.G., Kurz, M.B., 2004. Dynamic rescheduling that simultaneously considers efficiency and stability. Comput. Ind. Eng. 46, 1–15. https://doi.org/10.1016/j.cie.2003.09.007

Rao, S.S., 1987. Game theory approach for multiobjective structural optimization. Comput. Struct. 25, 119–127. https://doi.org/10.1016/0045-7949(87)90223-9

Ren Shan, X.Z., 2018. A framework for shopfloor material delivery based on real-time manufacturing big data. J. Ambient Intell. Humaniz. Comput. https://doi.org/10.1007/s12652-018-1017-7

Renna, P., 2010. Job shop scheduling by pheromone approach in a dynamic environment. Int. J. Comput. Integr. Manuf. 23, 412–424. https://doi.org/10.1080/09511921003642170

Rossi, A., 2014. Flexible job shop scheduling with sequence-dependent setup and transportation times by ant colony with reinforced pheromone relationships. Int. J. Prod. Econ. 153, 253–267. https://doi.org/10.1016/j.ijpe.2014.03.006

Sahin, C., Demirtas, M., Erol, R., Baykasoğlu, A., Kaplanoğlu, V., 2017. A multi-agent based approach to dynamic scheduling with flexible processing capabilities. J. Intell. Manuf. 28, 1827–1845. https://doi.org/10.1007/s10845-015-1069-x

Sharma, P., Jain, A., 2016. New setup-oriented dispatching rules for a stochastic dynamic job shop manufacturing system with sequence-dependent setup times. Concurr. Eng. Res. Appl. 24, 58–68. https://doi.org/10.1177/1063293X15599814

Shen, X.-N., Yao, X., 2015. Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems. Inf. Sci. (Ny). 298, 198–224. https://doi.org/10.1016/j.ins.2014.11.036

Shukla, O.J., Gunjan, S., Rajesh, K., Sujil, A., 2018. An agent-based architecture for production scheduling in dynamic job-shop manufacturing system. - Autom. 66, 492. https://doi.org/10.1515/auto-2017-0119

Sun, D.H., He, W., Zheng, L.J., Liao, X.Y., 2014. Scheduling flexible job shop problem subject to machine breakdown with game theory. Taylor and Francis Ltd., pp. 3858–3876. https://doi.org/10.1080/00207543.2013.784408

Tang, L., Liu, W., Liu, J., 2005. A neural network model and algorithm for the hybrid flow shop scheduling problem in a dynamic environment. J. Intell. Manuf. 16, 361–370. https://doi.org/10.1007/s10845-005-7029-0

Umar, U.A., Ariffin, M.K.A., Ismail, N., Tang, S.H., 2015. Hybrid multiobjective genetic algorithms for integrated dynamic scheduling and routing of jobs and automated-guided vehicle (AGV) in flexible manufacturing systems (FMS) environment. Int. J. Adv. Manuf. Technol. 81, 2123–2141. https://doi.org/10.1007/s00170-015-7329-2

Vieira, G.E., Herrmann, J.W., Lin, E., 2000. Predicting the performance of rescheduling strategies for parallel machine systems. J. Manuf. Syst. 19, 256–266. https://doi.org/10.1016/S0278-6125(01)80005-4

Wang, H., Jiang, Z., Wang, Y., Zhang, H., Wang, Y., 2018. A two-stage optimization method for energy-saving flexible job-shop scheduling based on energy dynamic characterization. J. Clean. Prod. 188, 575–588. https://doi.org/10.1016/j.jclepro.2018.03.254

Wang, J., Zhang, Y., Liu, Y., Wu, N., 2018. Multi-agent and Bargaining-game-based Real-time Scheduling for Internet of Things-enabled Flexible Job Shop. IEEE Internet Things J. PP, 1. https://doi.org/10.1109/JIOT.2018.2871346

Wang, L., Luo, C., Cai, J., 2017. A Variable Interval Rescheduling Strategy for Dynamic Flexible Job Shop Scheduling Problem by Improved Genetic Algorithm. J. Adv. Transp. 2017. https://doi.org/10.1155/2017/1527858

Wang, S. jin, Xi, L. feng, Zhou, B. hai, 2008. FBS-enhanced agent-based dynamic scheduling in FMS. Eng. Appl. Artif. Intell. 21, 644–657. https://doi.org/10.1016/j.engappai.2007.05.012

Wang, W., Yang, H., Zhang, Y., Xu, J., 2018. IoT-enabled real-time energy efficiency optimisation method for energy-intensive manufacturing enterprises. Int. J. Comput. Integr. Manuf. 31, 362–379. https://doi.org/10.1080/0951192X.2017.1337929

Wang, W., Zhang, Y., Zhong, R.Y., 2020. A proactive material handling method for CPS enabled shop-floor. Robot. Comput. Integr. Manuf. 61, 101849. https://doi.org/10.1016/j.rcim.2019.101849

Wu, X., Sun, Y., 2018. A green scheduling algorithm for flexible job shop with energy-saving measures. J. Clean. Prod. 172, 3249–3264. https://doi.org/10.1016/j.jclepro.2017.10.342

Xanthopoulos, A.S., Koulouriotis, D.E., Tourassis, V.D., Emiris, D.M., 2013. Intelligent controllers for bi-objective dynamic scheduling on a single machine with sequence-dependent setups. Appl. Soft Comput. J. 13, 4704–4717. https://doi.org/10.1016/j.asoc.2013.07.015

Yan, J., Li, L., Zhao, F., Zhang, F., Zhao, Q., 2016. A multi-level optimization approach for energy-efficient flexible flow shop scheduling. J. Clean. Prod. 137, 1543–1552. https://doi.org/10.1016/j.jclepro.2016.06.161

Yin, L., Li, X., Gao, L., Lu, C., Zhang, Z., 2017. A novel mathematical model and multi-objective method for the low-carbon flexible job shop scheduling problem. Sustain. Comput. Informatics Syst. 13, 15–30. https://doi.org/10.1016/j.suscom.2016.11.002

Zambrano Rey, G., Bekrar, A., Prabhu, V., Trentesaux, D., 2014. Coupling a genetic algorithm with the distributed arrival-time control for the JIT dynamic scheduling of flexible job-shops. Int. J. Prod. Res. 52, 3688–3709. https://doi.org/10.1080/00207543.2014.881575

Zhang, G., Shao, X., Li, P., Gao, L., 2009. An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. Comput. Ind. Eng. 56, 1309–1318. https://doi.org/10.1016/j.cie.2008.07.021

Zhang, G., Zhang, Y., Xu, X., Zhong, R.Y., 2018. An augmented Lagrangian coordination method for optimal allocation of cloud manufacturing services. J. Manuf. Syst. 48, 122–133. https://doi.org/10.1016/j.jmsy.2017.11.008

Zhang, G., Zhang, Y., Zhong, R.Y., Wu, Y., 2019. Extending augmented Lagrangian coordination for the optimal configuration of cloud-based smart manufacturing services with production capacity constraint. Robot. Comput. Integr. Manuf. 58, 21–32. https://doi.org/10.1016/j.rcim.2019.01.009

Zhang, L., Li, X., Gao, L., Zhang, G., 2016. Dynamic rescheduling in FMS that is simultaneously considering energy consumption and schedule efficiency. Int. J. Adv. Manuf. Technol. 87, 1387–1399. https://doi.org/10.1007/s00170-013-4867-3

Zhang, R., Chiong, R., 2016. Solving the energy-efficient job shop scheduling problem: A multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. J. Clean. Prod. 112, 3361–3375. https://doi.org/10.1016/j.jclepro.2015.09.097

Zhang, S., Wong, T.N., 2017. Flexible job-shop scheduling/rescheduling in dynamic environment: a hybrid MAS/ACO approach. Int. J. Prod. Res. 55, 3173–3196. https://doi.org/10.1080/00207543.2016.1267414

Zhang, Y., Liu, S., Liu, Y., Yang, H., Li, M., Huisingh, D., Wang, L., 2018. The 'Internet of Things' enabled real-time scheduling for remanufacturing of automobile engines. J. Clean. Prod. 185, 562–575. https://doi.org/10.1016/j.jclepro.2018.02.061

Zhang, Y., Wang, J., Liu, S., Qian, C., 2017a. Game Theory Based Real-Time Shop Floor Scheduling Strategy and Method for Cloud Manufacturing. Int. J. Intell. Syst. 32, 437–463. https://doi.org/10.1002/int.21868

Zhang, Y., Wang, J., Liu, Y., 2017b. Game theory based real-time multi-objective flexible job shop scheduling considering environmental impact. J. Clean. Prod. 167, 665–679. https://doi.org/10.1016/j.jclepro.2017.08.068

Zhang, Z., Tang, R., Peng, T., Tao, L., Jia, S., 2016. A method for minimizing the energy consumption of machining system: integration of process planning and scheduling. J. Clean. Prod. https://doi.org/10.1016/j.jclepro.2016.03.101

Zhao, F., Hong, Y., Yu, D., Yang, Y., Zhang, Q., 2010. A hybrid particle swarm optimisation algorithm and fuzzy logic for process planning and production scheduling integration in holonic manufacturing systems. Int. J. Comput. Integr. Manuf. 23, 20–39. https://doi.org/10.1080/09511920903207472

Zhou, G., Jiang, P., Huang, G.Q., 2009. A game-theory approach for job scheduling in networked manufacturing. Int. J. Adv. Manuf. Technol. 41, 972–985. https://doi.org/10.1007/s00170-008-1539-9

Highlights

- Infinitely repeated game method was used to reduce real-time scheduling's complexity.
- Production planning layer was used to generate new scheduling as a production planning.
- Real-time scheduling layer was used to finish operations from production planning layer.
- Applying real-time data can timely reduce the influence of exceptions.