IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 26, 2019, accepted August 20, 2019, date of publication August 30, 2019, date of current version September 16, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2938548

An Adaptive Real-Time Scheduling Method for Flexible Job Shop Scheduling Problem With Combined Processing Constraint

HAIHUA ZHU, MING CHEN, ZEQUN ZHANG, AND DUNBING TANG
College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210000, China

Corresponding author: Haihua Zhu (zhuhh@nuaa.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant U1637211 and Grant 51805253, in part by the Fundamental Research Funds for the Central Universities under Grant 3082018NS2018035, and in part by the National Defense Basic Scientific Research Program of China under Grant JCKY2018605C003.

ABSTRACT Flexible job shop scheduling problem with combined processing constraint is a common scheduling problem in assembly manufacturing industry. However, traditional methods for classic flexible job shop scheduling problem (FJSP) cannot be directly applied. To address this problem, the concepts of 'combined processing constraint' and 'virtual operation' are studied and introduced to simplify and transform FJSP with combined processing constraint into FJSP. A Multi-agent system (MAS) for FJSP is used for fitting the requirement of building complex, flexible, robust and dynamic manufacturing scheduling. On this basis, a novel adaptive real-time scheduling method for MAS is further proposed for better adaptability and performance. This method solves the previously converted problem and conquers the shortcoming of poor performance of traditional single dispatching rule method in MAS. In this approach, the scheduling process is modeled as contextual bandit, so that each job agent can select the most suitable dispatching rules according to the environment state after learning to achieve scheduling optimization. The proposed method is compared with some common dispatching rules that have been widely used in MAS. Results illustrate the high performance of the proposed method in a simulated environment.

INDEX TERMS Multi-agent system, flexible job shop scheduling problem, combined processing constraint, contextual bandit.

I. INTRODUCTION

Flexible job shop scheduling problem (FJSP) is a fundamental problem in manufacturing. It's a generalization of the job shop scheduling problem (JSP) [1], which removes the limitation of the unique machine specified in each operation and concerns the processing flexibility [2]. In the past few decades, experts and scholars have done a lot of researches on FJSP and developed various solution methodologies [3]. In many literature, most researchers assumed that a machine cannot process more than one operation at the same time [4] and that it only needs to meet the conventional routing constraints. However, in many industries, in order to ensure the accuracy of assembly, several jobs must be processed simultaneously on the same machine (i.e. combined processing). Combined processing is a processing technology that clamps two or more parts in accordance with the assembly relationship by using the same reference and processes the related operations in one single chucking. This technology can assure the accuracy of assembly, reduce the difficulty of processing and improve the working efficiency. It's widely used in the production of high precision components, like the manufacturing of various molds and shell parts. The scheduling problem in these job shops is exactly FJSP with combined processing constraint. In this problem, some machines can process multiple operations of different jobs at the same time. Moreover, in addition to meeting the conventional routing constraints, it is also necessary to meet the combined processing constraint between different jobs. Therefore, FJSP with combined processing constraint is more complex than the classic FJSP.

Although FJSP with combined processing constraint widely exists in the real-world job shop, there is few literature about this problem. Xiong *et al.* [5] and Yi *et al.* [6] have done

The associate editor coordinating the review of this article and approving it for publication was Weiguo Xia.

some research on JSP with combined processing constraint, but it just used a single dispatching rule method and did not extend to FJSP. In this article, by introducing the definition of 'combined processing constraint' and 'virtual operation', FJSP with combined processing constraint can be simplified and transformed into classic FJSP. In this way, it can be solved by methods for the classic FJSP.

In the real-world scheduling system, centralized approaches for FJSP, like genetic algorithm (GA), simulated annealing (SA) and tabu search (TS) are inefficient and impractical for solving large-sized problems owing to the increased computation time requirement [7]. Instead, MAS approach can solve this problem well because of its instantaneity, flexibility, reliability, adaptability and reconfigurability [8]. However, the traditional MAS approach uses only a single dispatching rule, ignoring the impact of system environment changes on the selection of dispatching rules, resulting in poor scheduling performance. In order to overcome the disadvantages of the traditional MAS approach, various methodologies have been developed. Liu [9] proposed a method of composite dispatching rules using analytic hierarchy process (AHP) to improve the performance of scheduling, but subjective experience is required. Liu [10] proposed an adaptive real-time scheduling method, which used Q learning algorithm to optimize the rule selection. Qu [11] used a reinforcement learning (RL) method to adaptively generate dispatching rules based on system state. Kacem *et al.* [12] investigated the combination schemes for evolving rule ensembles using genetic programming for dynamic job shop scheduling. Trentesaux [13] proposed four new dispatching rules by performing a simulation-based analysis for dynamic job shop scheduling. However, these methods can only be used for JSP.

In job shop scheduling, the dispatching rules suitable for each job are changing with different times and system states. FJSP is more complex than JSP, so the system states and the corresponding dispatching rules are more variable in the scheduling process. If only a single dispatching rule is specified globally, this variation is ignored and the scheduling performance will be inevitably poor. In this article, the scheduling process of job agent in FJSP consists of two stages [14], [15]: machine selection and buffer job sequencing, which are modeled as contextual bandit (CB) in RL. Through continuous trial-and-error learning, each job agent can select the most suitable machine selection rules and buffer job sequencing rules according to its experiences of the environment. Thus the adaptability of the job agent and the overall performance can be improved.

The rest of the paper is organized as follows: In Section 2 the flexible job shop scheduling problem with combined processing constraint is defined in detail and then the proposed method is explained in Section 3. Simulation experiments are presented in Section 4 and the conclusion is given in Section 5.

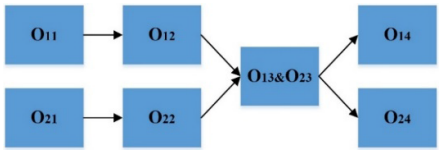


FIGURE 1. Combined processing.

II. FLEXIBLE JOB SHOP SCHEDULING PROBLEM

This paper is to evaluate an adaptive real-time scheduling method for FJSP with combined processing constraint. This problem can be stated as follows [16]–[18]:

(1) The jobs arrive in batches and the number of jobs arriving in each batch is random.
(2) The jobs needing combined processing can be processed only when the combined processing constraint (see section 2.1) is satisfied.
(3) Each operation may be executed on a set of alternative machines.
(4) The arrival time and type of a job are not known until the jobs arrives.
(5) Each machine can perform only one ordinary job or multiple jobs with operations needing combined processing at a time.
(6) Set-up time is included in the processing time and transportation time is not considered.
(7) A job, once taken up for processing on a machine, should be completed before another job is taken.

A. COMBINED PROCESSING CONSTRAINT

In the traditional scheduling problem, it is generally assumed that one machine can only process one job at the same time, which is in line with the situation of the ordinary job shop. However, in many industries, there are assembly relationships between certain jobs. If these jobs are separately processed, it will be difficult to guarantee the assembly precision. Therefore, in order to ensure the accuracy of assembly, several jobs must be processed simultaneously on one machine, i.e. combined processing. Combined processing is illustrated in Fig. 1. In this figure, O_{ij} denotes the jth operation of job J_i and J_i is the ith job to be processed. The first two operations of J_1, J_2 are processed separately and then the third operations of J_1, J_2 (O_{13}, O_{23}) need to be processed on one machine at the same time. Moreover, O_{14} and O_{24} can start processing only after O_{13} and O_{23} are finished. According to the description above, the definition of combined processing constraint can be given as follows:

Definition 1: Combined processing constraint. Different operations of two or more jobs must be processed simultaneously on one machine and the subsequent operations can be processed only after all these operations are finished.

B. PERFORMANCE MEASURE

The aim of scheduling process is to find the best process plan for a given job, the best machine for each operation and the

best sequence of operations on each machine with respect to the given performance measure. In this article, makespan is considered as the performance measure.

III. PROPOSED APPROACH

A. VIRTUAL OPERATION

In the traditional FJSP, each time only one operation of a job needs to be scheduled. So the job agent can make the unique decision for the scheduling. However, in FJSP with combined processing constraint, two operations of two different jobs need to be scheduled together when they need combined processing and meet the combined processing constraint. If the two job agents both have the ability of decision for this scheduling, in the early stage, different scheduling decisions will generate due to the randomness of the algorithm exploration process, thus causing divergence. In order to solve this divergence and make the scheduling decision of the operations needing combined processing be unique, the technology of the virtual operation is used in this paper. In the process of scheduling, the two operations are treated as one virtual operation. Since the composition of the components and the job number are unique, the operation with a smaller job number is used as the master operation and is responsible for making scheduling decision for the virtual operation. The other operation is auxiliary and only records the information related to itself. The concept of 'virtual operation' is defined as follows:

Definition 2: Virtual operation. Virtual operation consists of operations that require combined processing and it is scheduled like an ordinary operation. The operation with a small job number is the master operation, and it is responsible for the decision of the virtual operation scheduling. The other operations are auxiliary and only record information.

B. SOLUTION FOR COMBINED PROCESSING CONSTRAINT

FJSP with combined processing constraint is much more complex than classic FJSP and cannot be solved by methods for FJSP directly. However, in MAS approach, the scheduling is performed only according to the real-time information of the job shop and only one operation is scheduled at a time. Therefore, as shown in Fig. 2, by taking advantage of this feature and introducing the definition of 'virtual operation', FJSP with combined processing constraint can be simplified and transformed into classic FJSP.

The steps for solving combined processing constraint are described as follows:

(1) If there is a job to be scheduled, check whether it needs combined processing or not.
(2) If not need combined processing, schedule it directly; otherwise, judge whether it meets the combined processing constraint and turn step (3).
(3) If the combined processing constraint is met, the operations requiring combined processing are combined into one virtual operation for scheduling; otherwise, wait for the release of other operations and turn step (3).

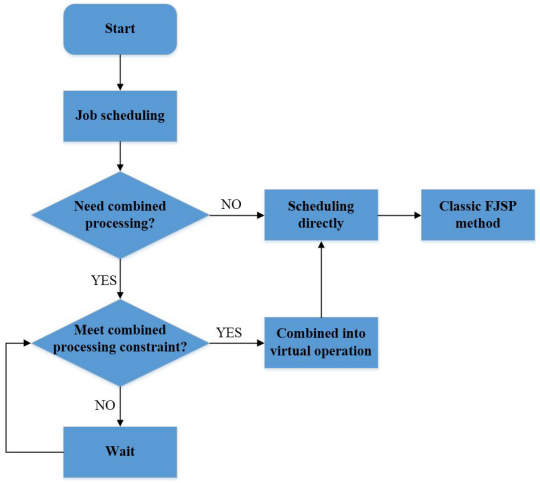


FIGURE 2. Solution for combined processing constraint.

In this way, FJSP with combined processing constraint can be simplified and transformed into classic FJSP and then solved by the methods for the classic FJSP.

C. ADAPTIVE REAL-TIME SCHEDULING METHOD

Today, as technology continues to evolve, the agility, flexibility and robustness of manufacturing system need to be improved to remain competitive [19]. MAS approach is based on the idea that several distributed agents can cooperate and coordinate in order to obtain globally optimal performances [20]. Therefore, it has characteristics of flexibility, reliability, adaptability and reconfigurability and is a method that meets the requirements of modern manufacturing system. However, the traditional MAS approach uses only a single dispatching rule, ignoring the impact of system environment changes on the selection of dispatching rules, resulting in poor scheduling performance. This paper uses Contextual bandit (CB) in RL to model the scheduling process, so that each job agent can select the best dispatching rules according to the environment state after trial-and-error learning. In this way, the adaptability of the job agent and the overall performance can be improved.

1) CONTEXTUAL BANDIT

CB is a special reinforcement learning model, which contains only one state per episode and only affects immediate reward [21]. The CB can be described as a tuple {S, A, R}, in which S is an unknown distribution over states (or 'contexts'), A is a known set of actions (or 'arms') and R is an unknown probability distribution over rewards. At each step t, agent selects an action a_t based on the environment state s_t and gets a reward r_t. The rewards in CB depend on the context information provided at each time slot [22]. This also means that varying environment state is quantized as contextual information to assist the decision making in a context-related, highly dynamic, complex system [23]. The purpose of agent is to achieve the best strategy

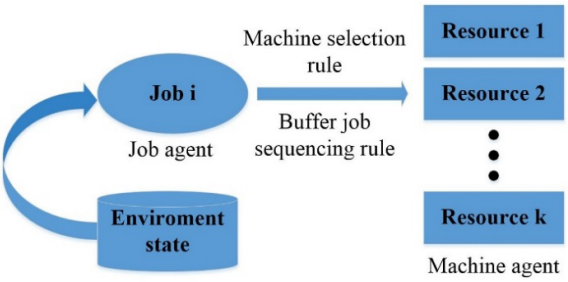


FIGURE 3. Decisional process of job agent.

(the state-to-action mapping) by continuous trial-and-error learning to maximize the cumulative reward.

2) CONTEXTUAL BANDIT FORMULATION FOR DECISIONAL PROCESS

As shown in Fig. 3, in MAS approach, the decisional process of job agent is a single-step decision and it generally consists of two stages: machine selection and buffer job sequencing, which need to select corresponding machine selection rules and buffer job sequencing rules. This is consistent with the description of CB and can be naturally modeled as CB problem. The contextual bandit formulation for decisional process is illustrated as follows:

a: STATE SPACE

The state space is composed of corresponding state features. When the job agent needs scheduling, it will collect state feature information and make decisions in real time. The state features selected in this paper include the number of every kind of operations scheduled at the same time, the number of jobs in the queue of available machines, the sum of processing time of waiting operations in the queue of available machines and the processing time of the considered operation on available machines.

b: ACTION SPACE

The action space consists of a combination of machine selection rules and buffer job sequencing rules. The machine selection rules are: Shortest Queue (SQ), Less Queued Element (LQE) and Shortest Processing Time (SPT). Buffer job sequencing rules are: First In First Out (FIFO), Shortest Job First (SJF) and Last In First Out (LIFO). Therefore, the action space includes a total of nine actions: SQ + FIFO, SQ + SJF, SQ + LIFO, LQE + FIFO, LQE + SJF, LQE + LIFO, SPT + FIFO, SPT + SJF, SPT + LIFO.

c: REWARD

After a decision making, the mean waiting time (MWT) of all jobs at time t is calculated and compared to MWT at time $t-1$. The corresponding reward is obtained by subtracting the current MWT from MWT at time $t-1$

$$MWT_t = \frac{\sum_{j=1}^{n} WT_{j,t}}{n} \tag{1}$$

$$r_t = MWT_{t-1} - MWT_t \tag{2}$$

where $WT_{j,t}$ is the remaining processing time of job j at time t; n is the total number of jobs.

3) SELECTION POLICY

In this paper, a CB policy, namely, LinUCB, is used to achieve the best rule selection policy. LinUCB is a CB algorithm that uses a linear model to approximate the relationship between the expected reward of each action and the state features. At each step t, the job agent obtains the state feature vector $x_{t,a} \in \mathbb{R}^d$ of action a through negotiation and the expected reward can be calculated by

$$E\left[r_{t,a} \middle| x_{t,a}\right] = x_{t,a}^T \theta_a^* \tag{3}$$

where $r_{t,a}$ is the expected reward of action a at step t and θ_a^* is the true coefficient vector of action a.

Algorithm 1 LinUCB With Online Updates [24]

Input and Initialize: $\alpha \in \mathbb{R}_+$
for $t = 1 \to T$ **do**
  Observe features of all actions $x_{t,a} \in \mathbb{R}^d$, $a \in A$
  for all $a \in A$ **do**
  if a is new **then**
  $A_a \leftarrow I_d$ (d-dimensional identity matrix)
  $b_a \leftarrow 0_{d\times 1}$ (d-dimensional zero vector)
  end if
  $\hat{\theta}_a \leftarrow A_a^{-1} b_a$
  $\hat{\mu}_a \leftarrow \hat{\theta}_a^T x_{t,a} + \alpha\sqrt{x_{t,a}^T A_a^{-1} x_{t,a}}$
end for
  Choose action $a_t = \arg\max_{a\in A} \hat{\mu}_a$ and observe a real-valued reward r_{t,a_t}
  $A_{a_t} \leftarrow A_{a_t} + x_{t,a_t} x_{t,a_t}^T$
  $b_{a_t} \leftarrow b_{a_t} + r_{t,a_t} x_{t,a_t}$
end for

The coefficients of each action can be estimated based on the agent's historical decision making experience. Let $G_a \in \mathbb{R}^{m\times d}$ and $c_a \in \mathbb{R}^m$ be design matrix at trial t, where each row of G_a represents the feature vector input before and each row of c_a represents the corresponding reward. Then a closed-form estimator of θ_a^* is obtained by ridge regression [24]:

$$\hat{\theta}_a = \left(G_a^T G_a + I_d\right)^{-1} b_a \tag{4}$$

where I_d is a d- dimensional identity matrix and $b_a = G_a^T c_a$.

In addition, in order to fully explore various actions, the LinUCB algorithm uses confidence interval for rule selection and always selects the action with the highest upper confidence bound. That is, at each step t, choose

$$a_t := \arg\max_{a\in A} \hat{\mu}_a = \arg\max_{a\in A} \left(x_{t,a}^T \hat{\theta}_a + \hat{\sigma}_a\right) \tag{5}$$

and $\hat{\sigma}_a = \alpha\sqrt{x_{t,a}^T A_a x_{t,a}}$, where $A_a := G_a^T G_a + I_d$ and $\alpha$is a parameter that controls the degree of exploration. The detailed description of LinUCB can be described as follows.

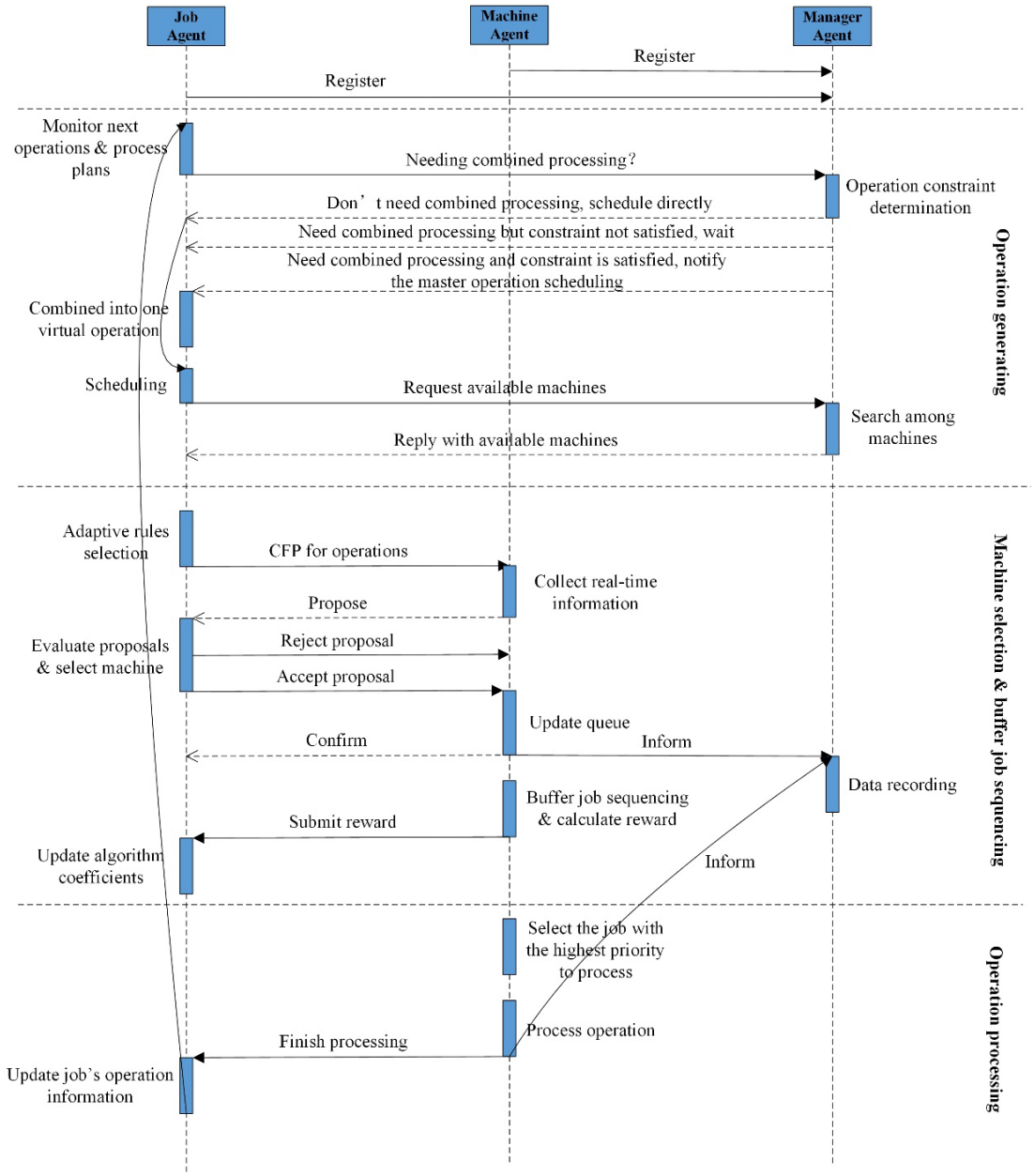


FIGURE 4. The UML sequence diagram of adaptive real-time scheduling method.

4) ADAPTIVE REAL-TIME SCHEDULING STRATEGY

The novel adaptive real-time scheduling strategy proposed in this paper is illustrated in Fig. 4. The optimal scheduling scheme with the makespan indicator can be achieved by the cooperation among the manager agent, the job agent and the machine agent. The main functions of each agent are as follows:

(1) Manager agent: There is only one manager agent in a job shop. This agent has the role of registering agents and recording the current information and status of all agents. It can determine whether or not a job agent can be scheduled and provide feasible process plans for each job agent.

(2) Job agent: Each job will have its own job agent. This agent has the role of selecting the most suitable machine selection rule and buffer job sequencing rule according to its experiences of the environment. It can determine the final process plan by these selected rules and the negotiation with the machine agent.

(3) Machine agent: Each machine will have its own machine agent. The main role of this agent is to negotiate with the job agent and determine its own processing order.

The detailed steps for negotiation between agents can be stated as follows:

TABLE 1. Information of jobs.

Jobs	Operations	Processing time									
		M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
J_1	O_{11}	10	8	—	—	—	—	—	—	—	—
	O_{12}	—	—	—	—	—	—	—	—	5	6
	$O_{13}(O_{24})$	—	—	—	—	4	6	—	—	—	—
J_2	O_{21}	8	10	—	—	—	—	—	—	—	—
	O_{22}	—	—	—	—	—	—	—	—	4	5
	O_{23}	—	—	—	—	—	—	5	3	—	—
	$O_{24}(O_{13})$	—	—	—	—	4	6	—	—	—	—
J_3	O_{31}	12	9	—	—	—	—	—	—	—	—
	$O_{32}(O_{42})$	—	—	—	—	8	6	—	—	—	—
	O_{33}	—	—	—	—	—	—	3	6	—	—
	O_{34}	—	—	—	—	—	—	—	—	8	6
J_4	O_{41}	—	—	5	8	—	—	—	—	—	—
	$O_{42}(O_{32})$	—	—	—	—	8	6	—	—	—	—
	O_{43}	—	—	—	—	—	—	—	—	7	9
	O_{44}	—	—	—	—	—	—	4	3	—	—
J_5	O_{51}	—	—	6	8	—	—	—	—	—	—
	O_{52}	9	5	—	—	—	—	—	—	—	—
	O_{53}	—	—	—	—	2	3	—	—	—	—
	$O_{54}(O_{63})$	—	—	—	—	—	—	—	—	7	5
J_6	O_{61}	—	—	6	4	—	—	—	—	—	—
	O_{62}	—	—	—	—	10	12	—	—	—	—
	$O_{63}(O_{54})$	—	—	—	—	—	—	—	—	7	5
J_7	O_{71}	—	—	10	8	—	—	—	—	—	—
	O_{72}	4	6	—	—	—	—	—	—	—	—
	$O_{73}(O_{82})$	—	—	—	—	—	—	6	8	—	—
	O_{74}	—	—	—	—	3	7	—	—	—	—
	$O_{75}(O_{84})$	—	—	—	—	—	—	—	—	6	8
J_8	O_{81}	—	—	13	11	—	—	—	—	—	—
	$O_{82}(O_{73})$	—	—	—	—	—	—	6	8	—	—
	O_{83}	3	5	—	—	—	—	—	—	—	—
	$O_{84}(O_{75})$	—	—	—	—	—	—	—	—	6	8
J_9	O_{91}	9	6	—	—	—	—	—	—	—	—
	O_{92}	—	—	4	5	—	—	—	—	—	—
	O_{93}	—	—	—	—	—	—	8	7	—	—
	$O_{94}(O_{104})$	—	—	—	—	—	—	—	—	4	9
J_{10}	O_{101}	—	—	3	8	—	—	—	—	—	—
	O_{102}	—	—	—	—	9	4	—	—	—	—
	O_{103}	—	—	—	—	—	—	3	5	—	—
	$O_{104}(O_{94})$	—	—	—	—	—	—	—	—	4	9

(1) The machine agent and the arrived job agent are registered at the manager agent.

(2) When job agent starts to release available operations to the shop, the manager agent is notified to determine

TABLE 2. Batches of jobs selected.

Batch No	Jobs included
Batch 1	Job 1~10
Batch 2	Job 1~8
Batch 3	Job 1~6, 9, 10
Batch 4	Job 1, 2, 5, 6, 9, 10
Batch 5	Job 3~10

whether or not the combined processing is required and whether the combined processing constraint is satisfied if the combined processing is required.

(3) If the manager agent determines that combined processing is not required, the operation can be directly scheduled and turn step (4). Otherwise, it further determines whether the combined processing constraint is satisfied: if satisfied, the operations requiring combined processing are combined into one virtual operation and notify the master operation scheduling, and then turn step (3); otherwise, the operation will wait for the combined operations to be released and not be scheduled.

(4) The master job agent requests the manager agent for resources for specific processes; the manager agent then searches among all registered machine agents and provides feedback on the available machines for processing the requested operation.

(5) According to the state feature, the job agent calls LinUCB algorithm of the operation released to select the best rules.

(6) The job agent issues a call for proposal (CFP) for its operation to all possible machine agents.

(7) Each machine agent prepares a proposal according to machine status and statistics.

(8) The job agent evaluates all proposals according to the machine selection rule selected in step (5). Then it selects the machine with the best proposal and sends an 'accept' message to the corresponding machine agent.

(9) The machine agent receiving the 'accept' message confirms the proposal. If the machine is idle, the operation will be directly performed; otherwise, it will be placed into the machine's queue and the buffer job sequencing rule is used to prioritize the queue. Then the change of MWT is calculated and submitted as a reward to the job agent.

(10) The job agent updates the coefficients of the corresponding operation algorithm according to the received reward.

(11) When the machine finishes the processing of an operation, if all operations of the job are finished, the job will be stored in the warehouse and the job agent will be cancelled. Otherwise, it will release the next operation and repeat steps (2) to (8). After that, the idle machine will select the job with the highest priority from its queue to process.

IV. EXPERIMENTS

A. EXPERIMENTAL DATA

To investigate the impact of our method, a job shop with ten machines and ten different job types are considered. The Tab. 1 gives processing time of each job type, in which the operation with brackets needs combined processing and the combined operation is in parentheses. The number in the table is the processing time of the operation on the corresponding machine and '—' means that this operation cannot be processed on this machine.

In many manufacturing industries with combined processing needs, especially in order-oriented ones, jobs are often arrived in batches intermittently on the job shop. Thus, in the simulation experiment of this paper, the jobs arrive in batches and the number of jobs of each batch is randomly generated from 5 to 10. Moreover, the jobs are randomly selected from Tab. 1 and the jobs with combined processing relationship must be selected at the same time. In order to better compare the makespan indicator, 5 batches of randomly generated jobs (see in Tab. 2) are selected and the inter-arrival time is set to 5. Then, a performance comparison is performed between the proposed algorithm and several most common dispatching rules described in section 3.2.2.

B. SIMULATION RESULTS

Simulation starts with the 5 batches of randomly generated jobs and performance comparison is performed between 9 single dispatch rules (SQ + FIFO, SQ + SJF, SQ + LIFO, LQE + FIFO, LQE + SJF, LQE + LIFO, SPT + FIFO, SPT + SJF, SPT + LIFO) and the adaptive real-time scheduling method. The adaptive real-time scheduling method first learns 800 epochs and makespan per epoch is shown in Fig. 5. It can be observed that by continuous learning, makespan is gradually reduced and finally stabilized between 110 and 115. The average of makespans of the last 10 epochs is set as the result of our method and compared with the results of 9 single dispatch rules. As shown in Fig. 6, our method can achieve a better solution than the single dispatch rule method. The result of the method has more than 10% improvement even compared to the best rule SPT + FIFO of the 9 single dispatching rules on this simulation experiment.

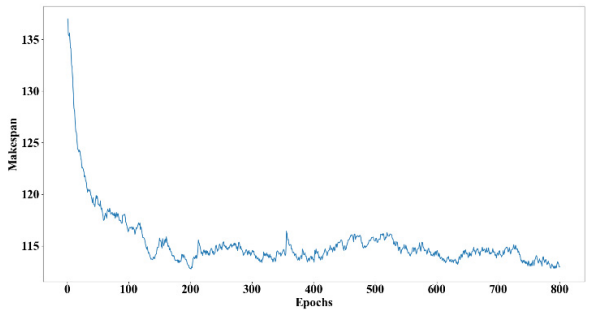


FIGURE 5. Makespan of each epoch.

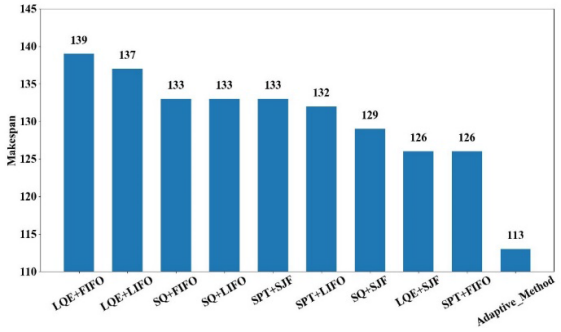


FIGURE 6. Comparison between various methods.

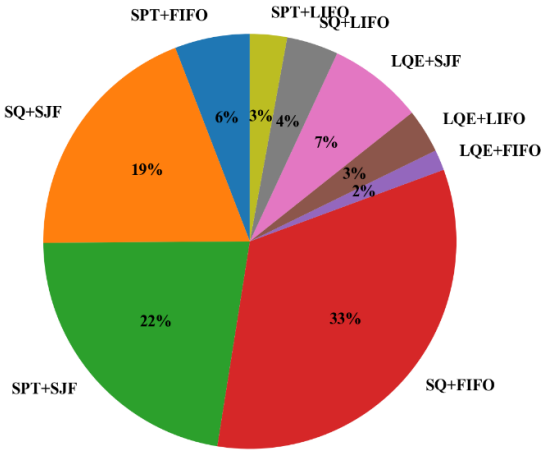


FIGURE 7. Dispatching rules distribution.

By this comparison, it is clearly shown that the proposed adaptive real-time scheduling method is effective. Furthermore, the distribution of various dispatching rules when using the adaptive scheduling method is represented in Fig. 7. It shows that the dispatching rules suitable for each job are changing with different times and system states.

V. CONCLUSION

FJSP with combined processing constraint is common in the manufacturing industry and is more complex than the classic FJSP. In this paper, by introducing the definition of 'combined processing constraint' and 'virtual operation', the problem can be successfully simplified and transformed into FJSP and the MAS approach for classic FJSP can be used directly. At the same time, a novel adaptive real-time scheduling method for MAS is proposed for this problem to overcome the shortcomings of traditional single dispatching rule method. In this method, contextual bandit in RL is used to model the scheduling process and the LinUCB algorithm is used for strategy learning. In this way, each job agent can select the most suitable dispatching rules according to the environment state after learning to achieve scheduling optimization. This approach significantly enhances the performance of the scheduling method. The proposed method is compared to some common dispatching rules using a simulated job shop. The results indicate that the performance of the proposed method is significantly better than those of the common dispatching rules.

In the future, to make the scheduling problem more realistic, this work will be further investigated with considering various dynamic disturbances, such as machine breakdowns, changes in release dates, order cancellation etc. The scheduling method could be more practical by taking into these factors.

REFERENCES

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

[2] R. Dong and W. P. He, "Hybrid genetic algorithm-ant colony optimization for FJSP solution," *Comput. Integr. Manuf. Syst.*, vol. 18, no. 11, pp. 2492–2501, 2012.

[3] D. Cinar, Y. Topcu, and J. A. Oliveira, "A taxonomy for the flexible job shop scheduling problem," in *Optimization, Control, and Applications in the Information Age* (Proceedings in Mathematics & Statistics). Macedonia, Greece: Springer, 2014.

[4] W. Xiang and H. P. Lee, "Ant colony intelligence in multi-agent dynamic manufacturing scheduling," *Eng. Appl. Artif. Intell.*, vol. 21, no. 1, pp. 73–85, 2008.

[5] H. Xiong, "Heuristic method for dynamic job shop scheduling problem with operation relativity," *J. Mech. Eng.*, vol. 42, no. 8, pp. 50–55, 2006.

[6] P. Yi, J. Li, and H. Xiong, "Heuristic method for job shop scheduling problem with combination processing in mould and die enterprise," *Forging Stamping Technol.*, vol. 34, no. 4, pp. 110–113, 2009.

[7] C. M. Joo and B. S. Kim, "Hybrid genetic algorithms with dispatching rules for unrelated parallel machine scheduling with setup time and production availability," *Comput. Ind. Eng.*, vol. 85, pp. 102–109, Jul. 2015.

[8] A. Rajabinasab and S. Mansour, "Dynamic flexible job shop scheduling with alternative process plans: An agent-based approach," *Int. J. Adv. Manuf. Technol.*, vol. 54, nos. 9–12, pp. 1091–1107, 2001.

[9] X. Liu, "Research on some critical issues about job shop real time scheduling," Ph.D. dissertation, Chongqing Univ., Chongqing, China, 2013.

[10] X. Liu, "Research on the job shop real-time scheduling based on adaptive dispatching rule," *Modular Mach. Tool Autom. Manuf. Technique*, vol. 2, pp. 157–160, 2014.

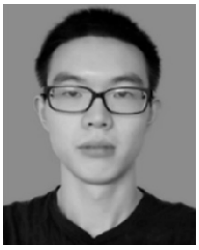
[11] S. Qu, J. Wang, and G. Shivani, "Learning adaptive dispatching rules for a manufacturing process system by using reinforcement learning approach," in *Proc. IEEE 21st Int. Conf. Emerg. Factory Automat.*, Berlin, Germany, Sep. 2016, pp. 1–8.

[12] I. Kacem, S. Hammadi, and P. Borne, "Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 32, no. 1, pp. 1–13, Feb. 2002.

[13] D. Trentesaux, C. Pach, A. Bekrar, Y. Sallez, T. Berger, T. Bonte, P. Leitão, and J. Barbosa, "Benchmarking flexible job-shop scheduling and control systems," *Control Eng. Pract.*, vol. 21, no. 9, pp. 1204–1225, 2013.

[14] W. Xiong and D. Fu, "A new immune multi-agent system for the flexible job shop scheduling problem," *J. Intell. Manuf.*, vol. 29, no. 4, pp. 857–873, 2018.
[15] J. Shahrabi, M. A. Adibi, and M. Mahootchi, "A reinforcement learning approach to parameter estimation in dynamic job shop scheduling," *Comput. Ind. Eng.*, vol. 110, pp. 75–82, Aug. 2017.
[16] M. Nouiri, A. Bekrar, and A. Jemai, "An effective and distributed particle swarm optimization algorithm for flexible job-shop scheduling problem," *J. Intell. Manuf.*, vol. 29, no. 3, pp. 603–615, Mar. 2018.
[17] P. Leitão, "Agent-based distributed manufacturing control: A state-of-the-art survey," *Eng. Appl. Artif. Intell.*, vol. 22, no. 7, pp. 979–991, 2009.
[18] G. Weiss, "Multiagent systems: A modern approach to distributed artificial intelligence," *Ai Mag.* vol. 14. no. 3, pp. 75–77, 2001.
[19] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. London, U.K.: MIT Press, 2017 p. 31.
[20] E. Schulz, E. Konstantinidis, and M. Speekenbrink, "Putting bandits into context: How function learning supports decision making," *J. Exp. Psychol., Learn., Memory, Cognition*, vol. 44, no. 6, pp. 927–943, 2017.
[21] X. Li, J. Liu, L. Yan, S. Han, and X. Guan, "Relay selection in underwater acoustic cooperative networks: A contextual bandit approach," *IEEE Commun. Lett.*, vol. 21, no. 2, pp. 382–385, Feb. 2017.
[22] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. 19th Int. Conf. World Wide Web*, Raleigh, NC, USA, Apr. 2010, pp. 661–670.

HAIHUA ZHU was born in Ningbo, China, in 1985. He received the B.S. degree in mechanical engineering and automation from the Nanjing University of Science and Technology, Nanjing, China, in 2008, and the Ph.D. degree from the Department of Industrial Engineering in the School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing, in 2013. He was supported by the China Scholarship Council. He has conducted his research with the School of Engineering, University of Greenwich, U.K., as a joint Ph.D. Student, from 2009 to 2011. Since 2013, he has been a Lecturer with the School of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing. His research interests include intelligent manufacturing systems and automations, and product lifecycle management.



MING CHEN received the B.S. degree in mechanical engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2017, where he is currently pursuing the M.S. degree in mechanical and electrical engineering.

ZEQUN ZHANG received the B.S. degree in mechanical engineering from the Nanjing University of Science and Technology, Nanjing, China, in 2013, and the M.S. degree in mechanical engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, in 2016, where he is currently pursuing the Ph.D. degree in mechanical engineering.

DUNBING TANG was born in Hunan, China, in 1972. He received the Ph.D. degree from the Nanjing University of Science and Technology (NUST), in 2000. Then, he spent two years on his Postdoctoral research with Tsinghua University, Beijing, and the City University of Hong Kong. He was supported by the renowned Alexander von Humboldt Foundation. He has conducted his research with RWTH Aachen University, Germany, as an Alexander von Humboldt Research Scientist, from July 2002 to February 2004. Thereafter, he moved to Cranfield University, U.K., as a Research Fellow. He was offered a full-time professorship. He joined the Nanjing University of Aeronautics and Astronautics, in December 2005. He has published over 100 academic articles. His research interests include intelligent manufacturing systems and automation, and manufacturing system modeling. He has conducted several research grants as a principal investigator (PI) or the Co-PI. His research outcome can be found in international high-quality academic journals such as the *International Journal of Production Research*, the *International Journal of Computer Integrated Manufacturing*, *IMechE Part B-Journal of Engineering Manufacture*, *Computers & Industrial Engineering*, *The International Journal of Advanced Manufacturing Technology*, *Computers in Industry*, the *Journal of Intelligent Manufacturing*, *Robotics & Computer Integrated Manufacturing*, and *Concurrent Engineering Research & Applications*.