Knowledge-guided DRL for Resource Scheduling in Customized and Personalized Production

Shulin Lan[1], Yinfei Jiang[1], Chen Yang[2*], Yingchao Wang[2], Xingshan Yao[2], Lihui Wang[3]
1. School of Economics and Management, University of Chinese Academy of Science, Beijing, China
2. School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China
3. Department of Production Engineering, KTH Royal Institute of Technology, Stockholm, Sweden
{*Corresponding author: yangchen666@bit.edu.cn}

***Abstract*—The manufacturing landscape has witnessed a paradigm shift towards multi-variety and small-batch production for the customized and personalized product (CPP). But this paradigm poses significant challenges for the cloud manufacturing system: 1) wired production machines cannot support the ultra-flexible resource allocation for the CPP job; 2) the scheduling model largely neglects the reconfiguration time of machines; 3) the intelligent scheduling method is difficult to learn the policy in the high-dimensional CPP solution space. To address these issues, we propose an edge-computing and wireless-connection based CPP manufacturing system framework which allows for the dynamic and ultra-flexible allocation of operations and resources. Then reconfiguration time is modelled in the optimization problem and a knowledge-guided deep reinforcement learning algorithm is proposed to effectively explore optimal CPP scheduling policy in the high dimensional solution space. The experimental results demonstrated that the proposed algorithm obtained better scheduling results than traditional scheduling rules, effectively balancing processing time and reconfiguration time, thereby minimizing the overall jobshop makespan.**

Keywords*—knowledge-guided deep reinforcement learning, customized and personalized production, ultra-flexible system, reconfigurable resource scheduling

I. INTRODUCTION

As products of choice become increasingly rich and exceed demand, consumers tend to purchase customized and personalized products (CPPs) that best meet individual needs [1]. Therefore, enterprises will compete to support the rapid fulfillment of customized and personalized jobs in an efficient manner.

CPP jobs contain multiple successive operations (shown in Fig.1.), each of which can be completed on a certain type of manufacturing machines. Since some machines are capable of processing two or more types of operations, they can be reconfigured for the target operations and relocated to busy stages to accelerate processing.

However, the current cloud manufacturing (CMfg) system cannot be reconfigured flexibly to handle a group of CPP jobs submitted by individual customers to the CMfg platform due to the following reasons.

Firstly, the existing wired manufacturing systems can only support mass production, which is difficult to support dynamic reconfiguration and scheduling of manufacturing machines. Secondly, the machine reconfiguration time can significantly affect the production efficiency, which are largely neglected in previous research. Finally, existing smart scheduling methods can not generate a stable and effective scheduling strategy due to uncertain reconfiguration time and dimensional expansion of the solution space. This calls for new system architecture, problem formulation and intelligent optimization methods to tackle such a complex and challenging problem.

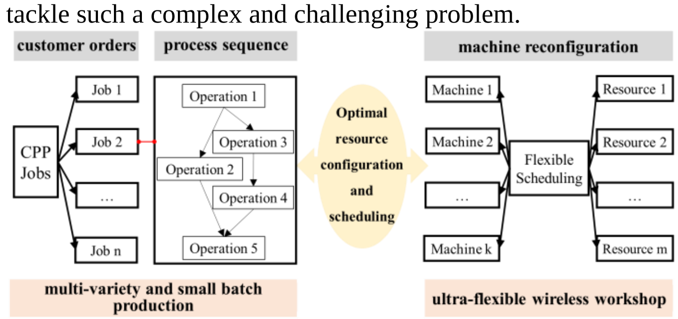


Fig. 1. Flexible scheduling in CPP production

As a result, the motivation of this study is to propose an intelligent resource scheduling and reconfiguration method with learning capacity in the ultra-flexible wireless workshop facing customized and personalized jobs and machine reconfiguration. The main contributions of this paper are highlighted as follows:

1) An edge-computing and wireless-connection CMfg framework is proposed to enable the dynamic and ultra-flexible reconfiguration of production machines and relative resource according to CPP jobs requirements.s

2) The CPPs reconfiguration time, CPP processing stages and reconfigurable machines are modelled in the optimization problem to better solve the CPP reconfiguration production and resource scheduling problem.

3) A knowledge-guided deep reinforcement learning algorithm is developed to quickly search stable solutions in the high-dimension CPP action space and significantly reduce the total makespan, considering the reconfiguration time.

The rest of this paper is as follows. In Section II, related work is presented. In Section III, the system framework is proposed. In Section IV, the details of model and algorithm are presented. In Section V, the results of the algorithm are shown. Finally, the conclusion of this experiment is drawn.

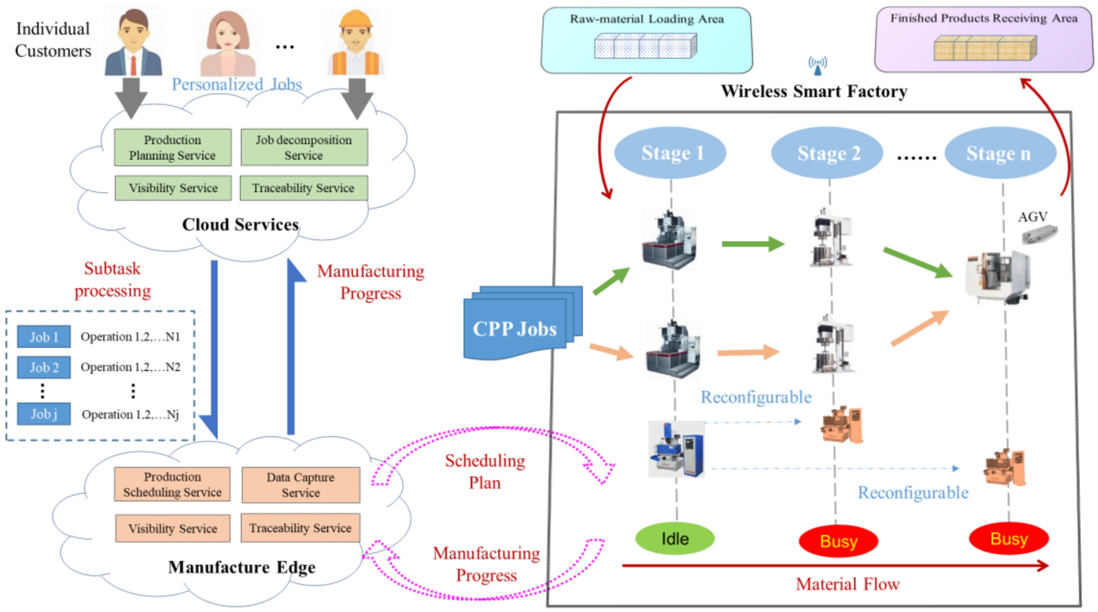


Fig. 2. An edge-cloud based cloud manufacturing system framework for mass customized and personalized production

II. Related Work

A. Flexible Manufacturing Systems for CPP production

The solution to flexible manufacturing can be categorized into two primary approaches. One involves breaking down the CPP jobs into precise components so that it can be scheduled by the flexible manufacturing system. Zhang et al. [2] optimized the resources in the CPP manufacturing mode to save the total production cost. Pang et al. [3] split the personalized jobs into multiple operations, which can be shared in different processing jobs. The other involves enhancing the performance of flexible manufacturing system, such as the capabilities in production manufacturing, including processing, scheduling and logistics [4-8]. Some other research focus on the improving flexibility, including the machine equipment and software components of manufacturing system [9-11].

These literatures about CPP production ignore the demand of dynamic job and resource allocation of each CPP jobs and do not obtain sufficiently effective solutions

B. Intelligent Production Scheduling Methods

The mainstream research about production scheduling solutions is intelligent algorithms. Liu and Yang [12,13] applied deep learning algorithms to study the scheduling policies of CMfg resources. Lei and Yu [14,15] respectively proposed intelligent model-based algorithms to tackle the issue of dynamic arrival of jobs during the scheduling process.

Current research has already encompassed various research domains of jobshop scheduling problem, but most of them neglects the characteristics of reconfigurable manufacturing systems and struggle in CPP high-dimension action space.

III. System Framework

This work proposed an edge-cloud based manufacturing system framework for personalized production. As shown in Fig. 2, the framework consists of the following components.

A. Personalized CMfg Platform

The CMfg platform can accept the personalized jobs submitted by distributed individuals. The jobs contain important personalized product parameters. The personalized jobs are decomposed into a lot of operations that can be processed in different stages of the workshop. It also provides visibility and traceability services for customers to obtain the progress of the manufacturing processes.

B. Edge Manufacturing Node

The edge manufacturing node (EMN) is an edge-computing based manufacturing service node that can accept personalized production jobs distributed from the CMfg platform, provide data processing and storage services for the shopfloor things and manage the production processes in the workshop. EMN collects the real-time status of the manufacturing things and makes smart decisions using those data and intelligent scheduling models and algorithms. According to the status of things and jobs, EMN can reconfigure the resources and schedule the jobs to the resources optimally.

C. Smart Wireless Connected Factory

All the elements in the factory are connected using wireless communication technologies such as 5G/6G, which is convenient to reconfigure and move the mobile manufacturing facilities to different stages, optimizing the manufacturing system. The stages are settled as the largest set of the processes required by different personalized jobs. A CPP job may only flow across some of the stages, thus easily leading to the unbalanced workload in different stages. In such cases, the multi-function machines in idle stages can be reconfigured and moved to busy stages, so as to reduce the total time of CPP jobs. Therefore, it is particularly important to make smart decisions on the reconfiguration and scheduling of production resources for CPP jobs.

IV. Problem Formulation

A. Model Construction

In the CMfg framework, the smart factory contains a set of CPP jobs $J=\{J_1,J_2,J_3,\cdots,J_n\}$ and reconfigurable machines $M=\{M_1,M_2,M_3,\cdots,M_m\}$, each job contains j stages, denoted as operations $O_i=\{O_{i1},O_{i2},O_{i3},\cdots,O_{ij}\}, i\in J$, the corresponding processing time of O_{ij} is P_{ij}. Reconfiguration time between stages and the processing speed of each stage varies with machines in the reconfigurable settings, so the reconfiguration time from operation *i* to *j* of machine k is Tr_{ijm} and the speed of operation *i* processed by machine m is Sp_{im}. In our study, we assume that the cost of processing and reconfiguration is zero and the only objective is to minimize the makespan $T = \max\{F_{ij}\}$, where F_{ij} is the completion time of operation O_{ij}. We formulate this problem as follows:

$$\min \max_{i\in 1\ldots n}\{T_i\} \tag{1}$$

$$s.t.\quad type_{ij} \in \bigcup_{r=1}^{s} \beta_{jr}^{i} \times Type_r \tag{2}$$

$$Ts_r \geq \sum_{r=1}^{s} \beta_{jr}^{i} \times Tt_{1\ r+1} \tag{3}$$

$$x_{jp}^{i} \times (ts_{ip} - te_{ij} - \sum_{u=1}^{s}\sum_{v=1}^{s} \beta_{ju}^{i}\beta_{pv}^{i} Tt_{uv}) \geq 0 \tag{4}$$

$$te_{ij} - ts_{ij} = \sum_{r=1}^{s} \beta_{jr}^{i} \times q_{ij} \times CT_r(type_{ij}) \times Sp_r\left(type_{ij}\right) \tag{5}$$

$$T_i = te_{ij} + \sum_{r=1}^{s} \beta_{jr}^{i} \times Tt_{r+1\ s+2} \tag{6}$$

$$\beta_{jr}^{i} \times \beta_{pr}^{l} \times \left(ts_{lp} - te_{ij}\right)(ts_{lp} - te_{ij} - \sum_{u=1}^{s}\sum_{v=1}^{s} \beta_{ju}^{i} \times \beta_{pv}^{i} \times TC_{uv}) \geq 0 \tag{7}$$

$$\beta_{jr}^{i} \times (ts_{ij} - Ts_r) \geq 0 \tag{8}$$

$$\beta_{jr}^{i} \times (te_{ij} - Te_r) \leq 0 \tag{9}$$

Constraint (2) ensures each operation should be assigned to accessible machine. Constraint (3) ensures first operation of each job is later that raw material transfer time. Constraint (4) ensures the previous process and the transfer of semi-finished products must be completed before the latter process can be started. Constraint (6) denotes the completion time of each job is the time when the product is transferred to finished product storage. Constraint (7) ensures the start time of the next process is later than the end time of the previous process plus the machine reconfiguration time.

B. MDP Formulation

The scheduling process is actually a series of consecutive decisions. At each decision step t (time 0 or when an operation is completed), the agent observes the current system state S_t and makes decision a_t. The operation-machine action is to allocate an unscheduled operation to an idle machine and start it from the current time, denoted as T(t). Then, the environment transits to the next decision step t + 1. The process iterates until all the operations are scheduled. The corresponding MDP is defined as follows.

State: The state is used to characterize the status of the system to guide the decision. Considering that the workshop environment consists of two components: machines and jobs, the state vector at time t therefore can be denoted as $S_t=\{N_t(k)\}, i\in\{J,M\}$, N_t is a six-element vector where $N_t(k)=(wb_t,at_t,nn_t,ut_t,co_t,cs_t)_i,\ \forall k\in\{J,M\}$.

(1) *wb* is working binary, which is represented as a binary number.
(2) *at* is available time, denotes the completion time of current job and it’s equal to the current time when the machine or job is idle.
(3) *nn* is number of neighbors, the neighbors of a job are machines that are capable of processing the job’s current operation, while the neighbors of a machine are the jobs it can process currently.
(4) *ut* denotes the utility of machine or job N, $ut(N)=working\ time(N)/current\ time$.
(5) *co* represents the number of current operation
(6) *cs* represents the current speed of N.

Action: This article combines the operation selection and the machine assignment as a composite decision , An action is defined as a feasible operation-machine pair and None type action $a_t=\left\{\left(o_{ij},M_k\right)\right\}\cup\left\{None\right\}$, where O_{ij} is eligible and M_k is idle and can process O_{ij} .The action vector is concatenated by operation and machine vector, $S_t^{'}=concat\left(N_t\left(J\right),N_t\left(M\right)\right)$.

Transition: State vector is dependent on the previous state and action, so the transition function will be $S_{t+1}\leftarrow S_t\,|\left(a_t=a\right)$. The new state S_{t+1} is the time when a new operation is completed after S_t.

Reward: Reward function is designed to estimate the action and optimize the policy. The reward function is commonly designed as the difference between the estimated completion times of S_t and S_{t+1}. Additionally, we have considered the impact of machine reconfiguration time to guide the agent in learning strategies that minimize reconfiguration time.

$$r\left(S_t,S_{t+1},a_t\right)=T\left(S_t\right)-T\left(S_{t+1}\right)-trantime\left(a_t\right)$$

Policy: A policy $\pi(a_t|S_t)$ defines a probability distribution over the action set for each state. Later in this section, we proposed a Knowledge-Guided DRL (KGDRL) algorithm that parameterizes π as a neural network and optimizes it with actor-critic based structure by maximizing the cumulative reward function.

C. Knowledge Guidance Structure

A knowledge-guided structure is deployed in the PPO-based deep reinforcement learning algorithm with PPO structure is proposed to solve this large-sized problem. The efficacy of knowledge-guidance structure in bolstering the efficacy of training results and in accelerating processing speed has been empirically validated [16,17].

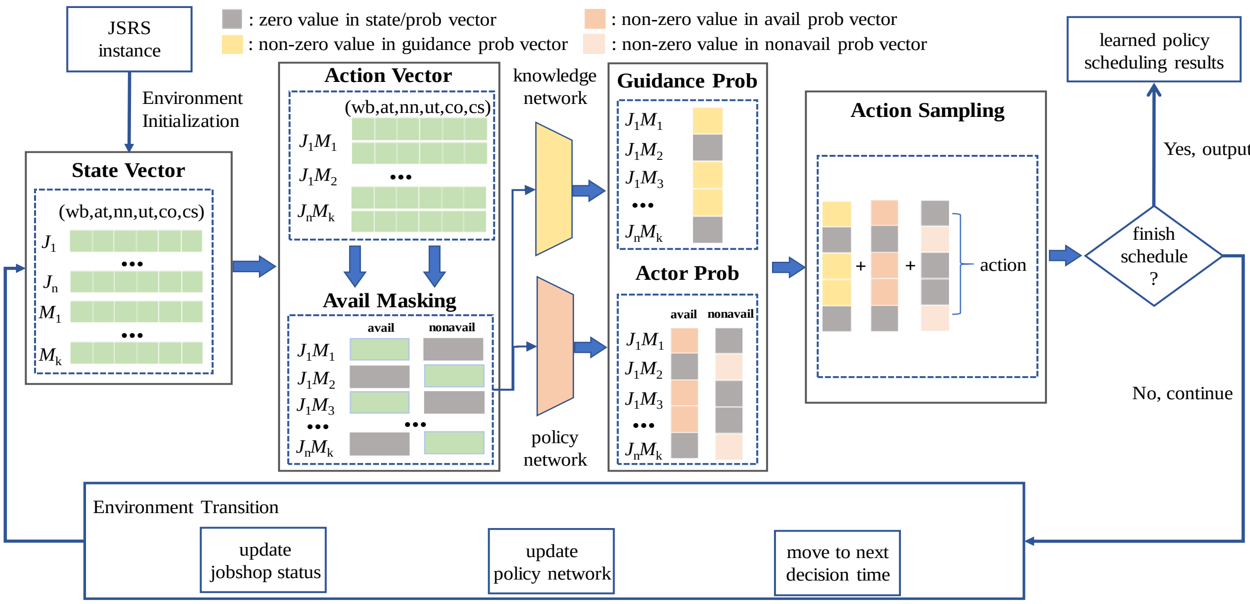


Fig. 3. Algorithm architecture

In the CPP scheduling problems, the makespan is a composition of both operation processing time and machine reconfiguration time. To procure the minimal makespan, a general knowledge guidance is proposed to minimize machine reconfiguration time during the entire scheduling process: IF the available action with shorter machine reconfiguration time, THEN the advice probability for selecting this action is comparatively higher. The general advice probability distribution of available action is shown as below, where *trantime* is machine reconfiguration time of available action.

$$S_1 = trantime(a_t) / \|trantime(a_t)\|_2$$

$$p_{a_t} = \alpha(e^{-S_1} / \|e^{-S_1}\|_1)$$

For each scheduling time T(t), the knowledge network output the guidance probability based on state vector and was combined with the actor probability from the policy network, then the learning agent will select the appropriate action based on the combined probability $p = p_{a_t} + p_{avail} + p_{nonavail}$

KGDRL uses Proximal Policy Optimization (PPO) structure for training, which deployed an actor-critic structure. Both actor and critic network are deployed with a L1-norm activation, the overall training structure is shown in Fig.3. As shown in Algorithm 1, the training is performed in I iterations and β independent instances during each iteration, we compute the advice probabilities based on each state and incorporate them into the probabilities outputted by the policy network, to optimize the sampling strategy of the agent.

Algorithm 1 : KGDRL

Initialize Actor network π_ω and Critic network V_ϕ with trainable parameters ω and ϕ

Initialize β independent instances

  For $iter = 1, 2, \ldots I$ **do :**

    For $b = 1, 2, \ldots \beta$ **do :**

      Initialize S_t based on instance b

      While S_t is not terminal **do :**

        Initialize mask vector F_{avail} and $F_{nonavail}$ based on a_t

          $F_{avail} = 1, F_{nonavail} = 0$ if $O_{ij} \in a_t$

        Compute action distribution based on policy network

          $p_{avail} = \pi_\omega\left(F_{avail}\left(O_{ij}\right)S_t\right), p_{nonavail} = \pi_\omega\left(F_{nonavail}\left(O_{ij}\right)S_t\right)$

        Compute general advice distribution $p_{advice} = p_{a_t}$

        Sample action a based on $p = p_{advice} + p_{avail} + p_{nonavail}$

        Receive reward r_t and next state S_{t+1}

          $S_t \leftarrow S_{t+1}$

      Compute GAE A_t based on r_t, S_{t+1} for each step

      Compute PPO loss Δ based on A_t

    Update network parameters ω and ϕ based on PPO loss Δ

Return

V. EXPERIMENTS

A. *Settings*

We adopted normal distribution to generate the respective processing time, reconfiguration times and processing speed of each instance, our generated instances including 6 sizes with varied jobs and machines, where n denotes job numbers and m denotes the machine numbers, we generated 10 instances each size for training, the detailed parameters are shown in Tab. 1.

We deployed one hidden layer in both the actor and the critic network. For training, the iteration $I = 1000$ and instances batch $\beta = 10$, min and max replay buffer is 128 and 2048 respectively. For PPO loss, the clip ratio and coefficients of entropy is 0.1 and 0.01 respectively. The PPO epochs are set to 15. These hyperparameters are empirically tuned on the 5 × 5 instance and fixed on the remaining sizes. The learned policies are compared to four widely used heuristic rules, including random method (randomly select one available action each time), SPT, FIFO, SSO and MWKR.

TABLE I. INSTANCE DETAILS

Size(n*m)	p_{ij}	Tr_{ijm}	Sp_{uv}
5*5, 10*5, 20*5, 5*10, 10*10, 20*10	N(100,0.1)	N(10,0,1)	N(1,0.1)

B. Performance on Instances

The production makespan of KGDRL and baselines are shown in Fig. 4. The training results of KGDRL outperform other rules in the majority of instances. Average makespan of 5 machines is better than that of 10 machines, indicating the superiority of the proposed algorithm in dealing with higher machine workload scheduling problems.

For each size, Tab.2 reports the average makespan of KGDRL and four rules. In large-scale instances (20*5 and 20*10), the scheduling outcomes of some rules are even inferior to those of random methods, which illustrates the difficulty in devising effective strategies for reconfiguration problem. But KGDRL can quickly converge in high-dimensional solution spaces, finding optimal scheduling strategies under the effective knowledge guidance.

C. Reconfiguration Analysis

As the key process of the CPP production, machine reconfiguration profoundly influences the future scheduling decisions of intelligent agents. As shown in Fig.5, machine 1 consecutively processed 5 operations after the second reconfiguration, indicating the pivotal role of machine reconfiguration in enhancing the CPP job scheduling efficiency.

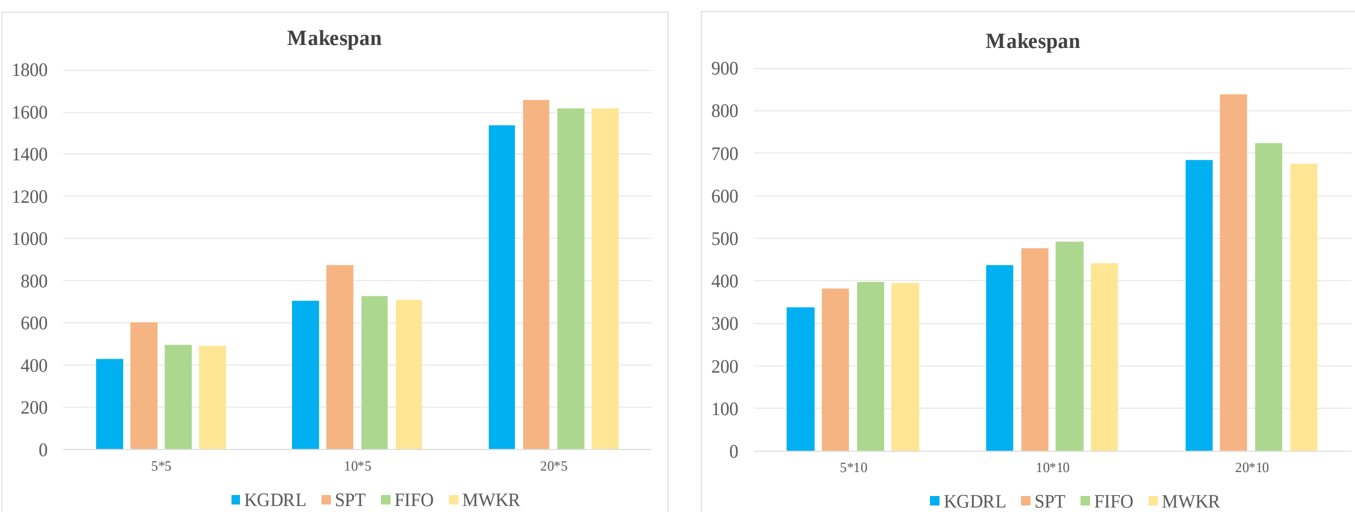
We further analysis the influence of the reconfiguration time on the overall makespan, the detail results of 10*5 instances are shown in Fig. 6. The scheduling strategy of KGDRL yields the shortest makespan and processing time, albeit at a shorter reconfiguration time compared to other commendable baselines thereby maximizing the operational efficiency of the jobshop.

TABLE II. PERFORMANCE OF PROPOSED ALGORITHM

Size		random	KGDRL	MWKR	FIFO	SPT	SSO
5*5	makespan	649.39	426.98	491.24	491.24	601.83	532.42
	enhance		34.25%	24.35%	24.35%	7.32%	18.01%
10*5	makespan	883.26	688.99	710.38	710.38	873.17	764.8
	enhance		21.99%	19.57%	19.57%	1.14%	13.41%
20*5	makespan	1735.4	1510.8	1618.4	1618.4	1658.3	1788.1
	enhance		12.94%	6.75%	6.75%	4.45%	-3.03%
5*10	makespan	483.15	333.82	394.41	394.41	381.88	415.84
	enhance		30.91%	18.37%	18.37%	20.96%	13.93%
10*10	makespan	514.53	411.87	440.48	440.48	475.81	480.89
	enhance		19.95%	14.39%	14.39%	7.53%	6.54%
20*10	makespan	778.59	683.52	676.1	676.1	839.03	750.93
	enhance		12.21%	13.16%	13.16%	-7.76%	3.55%

VI. CONCLUSIONS AND FUTURE RESEARCH

The CPP resource reconfiguration scheduling problem is studied and well resolved in this paper by the ultra-flexible reconfiguration manufacturing system and knowledge guided deep reinforcement learning. The main conclusions are summarized as follows. First, the edge-based ultra-flexible system is proposed to reconfigure production machines and relative resource to perform fast and flexible production according to CPP jobs requirements, thus can instantly analyze CPP jobs submitted by users and decomposed into operations to meet users' personalized needs. Second, the well-designed KGDRL is more efficient than traditional scheduling rules based on reconfiguration scheduling model. The idle machines can be reconfigured and effectively scheduled to busy stages to reduce the makespan of the production and promote efficient CPP production by increasing machine utilization.



(a) instances with various jobs and fixed 5 machines

(b) instances with fixed 5 jobs and various machines

Fig. 4. Mean Makespan of six sizes

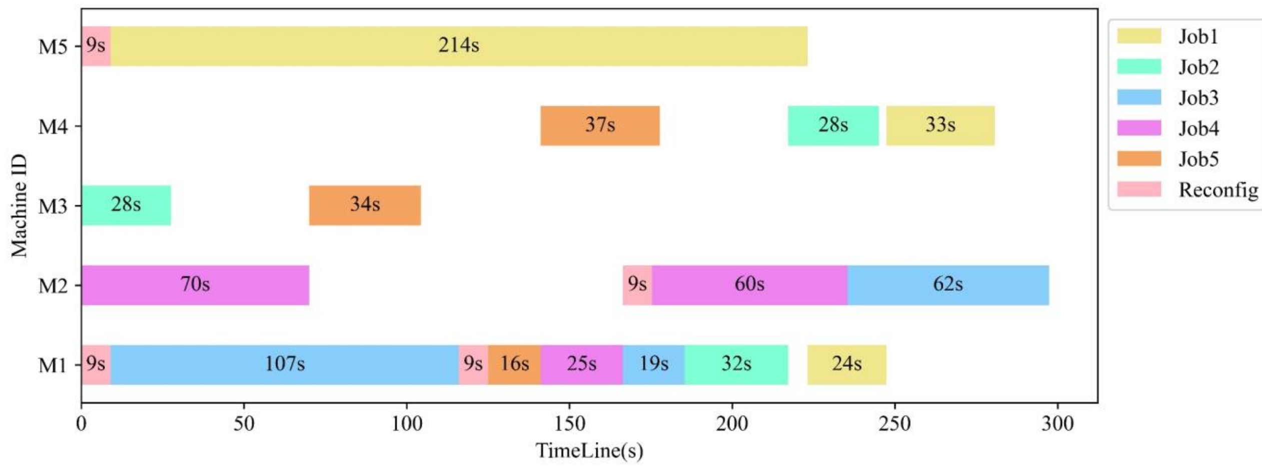


Fig. 5. Reconfiguration scheduling Gantt chart on 5*5 instance

In the future, digital twins can be introduced to ensure real-time monitoring of the processing workshop through human-machine collaboration and virtual-real connection mode, so the resource reconfiguration and scheduling with digital twins can be attempted against the uncertainties in cloud manufacturing.

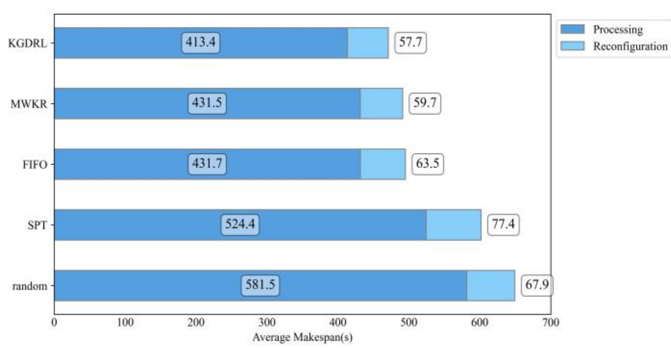


Fig. 6. Average makespan components on 5*5 instances

ACKNOWLEDGMENT

This work was supported by the National Key R&D Program of China under Grant 2021YFB1715700; in part by the National Natural Science Foundation of China under Grant 62103046, 72201266, 72192843, and 72192844; in part by the Fundamental Research Funds for the Central Universities under Grant E1E40805X2and 2023CX01020.

REFERENCES

[1] Yang, Chen, et al. "Towards product customization and personalization in IoT-enabled cloud manufacturing." Cluster Computing 20 (2017): 1717-1730.

[2] Zhang, Xianyu, Xinguo Ming, and Yuguang Bao. "A flexible smart manufacturing system in mass personalization manufacturing model based on multi-module-platform, multi-virtual-unit, and multi-production-line." Computers & Industrial Engineering 171 (2022): 108379.

[3] Pang, Shibao, et al. "Mass personalization-oriented integrated optimization of production task splitting and scheduling in a multi-stage flexible assembly shop." Computers & Industrial Engineering 162 (2021): 107736.

[4] Yang, Chen, Weiming Shen, and Xianbin Wang. "The internet of things in manufacturing: Key issues and potential applications." IEEE Systems, Man, and Cybernetics Magazine 4.1 (2018): 6-15.

[5] Yang, Chen, et al. "Metaverse: Architecture, Technologies, and Industrial Applications." 2023 IEEE 19th International Conference on Automation Science and Engineering (CASE). IEEE, 2023.

[6] Guo, Zhengang, et al. "CPS-based self-adaptive collaborative control for smart production-logistics systems." IEEE transactions on cybernetics 51.1 (2020): 188-198.

[7] Yang, Chen, et al. "Flexible resource scheduling for software-defined cloud manufacturing with edge computing." Engineering 22 (2023): 60-70.

[8] Lin, Chun-Cheng, et al. "Smart manufacturing scheduling with edge computing using multiclass deep Q network." IEEE Transactions on Industrial Informatics 15.7 (2019): 4276-4284.

[9] Koren, Yoram, et al. "Reconfigurable manufacturing systems." CIRP annals 48.2 (1999): 527-540.

[10] Leng, Jiewu, et al. "Digital twin-driven rapid reconfiguration of the automated manufacturing system via an open architecture model." Robotics and computer-integrated manufacturing 63 (2020): 101895.

[11] Lin, Ting Yu, et al. "Multi-centric management and optimized allocation of manufacturing resource and capability in cloud manufacturing system." Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture 231.12 (2017): 2159-2172.

[12] Liu, Yongkui, et al. "A framework for scheduling in cloud manufacturing with deep reinforcement learning." 2019 IEEE 17th international conference on industrial informatics (INDIN). Vol. 1. IEEE, 2019.

[13] Yang, Chen, et al. "Cloud-edge-device collaboration mechanisms of deep learning models for smart robots in mass personalization." Robotics and Computer-Integrated Manufacturing 77 (2022): 102351.

[14] Lei, Kun, et al. "Large-scale dynamic scheduling for flexible job-shop with random arrivals of new jobs by hierarchical reinforcement learning." IEEE Transactions on Industrial Informatics (2023).

[15] Yu, Weiwei, Li Zhang, and Ning Ge. "An adaptive multiobjective evolutionary algorithm for dynamic multiobjective flexible scheduling problem." International Journal of Intelligent Systems 37.12 (2022): 12335-12366.

[16] Yu, Yuanqiang, et al. "Accelerating deep reinforcement learning via knowledge-guided policy network." Autonomous Agents and Multi-Agent Systems 37.1 (2023): 17.

[17] Chen, Xiaocong, et al. "Knowledge-guided deep reinforcement learning for interactive recommendation." 2020 International Joint Conference on Neural Networks (IJCNN). IEEE, 2020.