IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON EDGE INTELLIGENCE FOR INTERNET OF THINGS

Received December 14, 2020, accepted January 5, 2021, date of publication January 8, 2021, date of current version January 20, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3050238

A Thing-Edge-Cloud Collaborative Computing Decision-Making Method for Personalized Customization Production

CHUN JIANG AND JIAFU WAN, (Member, IEEE)

Guangdong Provincial Key Laboratory of Technique and Equipment for Macromolecular Advanced Manufacturing, South China University of Technology, Guangzhou 510641, China

Corresponding author: Jiafu Wan (mejwan@scut.edu.cn)

This work was supported in part by the Key Areas Research and Development Program of Guangdong Province, China, under Grant 2019B010150002 and Grant 2019B090919002, and in part by the Key Program of National Natural Science Foundation of China under Grant U1801264.

ABSTRACT With the development of Industry 4.0 and cloud computing technology, personalized customization as a new production mode is showing a trend of rapid development. Personalized customization has the characteristics of order-driven production, strict processing times, high dynamic external conditions, and large flexibility in the production process, all of which bring more uncertainty to the production system and great challenges to the edge computing processing of related tasks in personalized customization production. Aiming at the above problems, a thing-edge-cloud collaborative computing decision-making (TCCD) method in customized production is proposed. First, the architecture of a personalized customized production system used for implementing the TCCD method is presented. Then, according to the number and type of products in the customer order received from the private cloud platform, the customer's personalized customized order is dynamically divided. Subsequently, a task priority sorting algorithm is proposed to optimize the waiting time of all tasks involved in the order. Furthermore, a discrete particle swarm algorithm is proposed to optimize the average execution time of all tasks and equipment utilization decision-making options (thing-edge collaborative computing, edge-edge collaborative computing, or edge-cloud collaborative computing). Finally, the effectiveness of the proposed TCCD method is verified by using the prototype platform of personalized product packaging intelligent production line with the same process flow.

INDEX TERMS
Edge collaborative computing, decision-making, Industry 4.0, customized production.

I. INTRODUCTION

Recently, new information technologies, such as artificial intelligence [1]–[3], cloud computing [4]–[6] and Internet of Things (IoT) [7]–[9], have emerged one after another, which have enlightened upgrades to the manufacturing industry. Nowadays, customer-needs show a trend of diversity and individualization, and the personalized customization production mode has gradually become mainstream. With the development of personalized and customized production, the demand for high-quality production is increasing, and the requirements for business delay, privacy, and security indicators have been further upgraded [10], [11]. The overall operation presents the trend of refinement, flexibility, and intelligence, which requires not only the overall operation of cloud computing but also the local real-time decision-making function of edge computing [12]–[14]. Edge computing is an open platform with a network, computing, storage, application, and other functions. It is located at the edge of the network near the device or data source and provides intelligent services to meet the key requirements of intelligent manufacturing, agile connection, real-time processing, data cleansing, and privacy protection [15]. Edge computing provides a mechanism for interconnection and intercommunication between devices, operational technology (OT) systems, and information technology (IT) systems as well as real-time data collection, aggregation, storage, and analysis mechanisms deployed in the manufacturing site, which can quickly and easily achieve integration of OT and IT [16]–[20]. However, in personalized customization production, higher requirements are put forward to shorten the average execution time of all the tasks related to production and make full

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

use of the equipment computing resources. The collaborative computing of the edge, cloud, and manufacturing equipment can effectively solve these problems.

In recent years, many scholars have analyzed and studied the problem of edge computing and edge computing task offloading decision-making in industrial IoT (IIoT), and some achievements have been made. Ma *et al*. [21] proposed a production system scheduling framework under the edge-cloud collaborative paradigm based on the dynamic fluctuation of orders under personalized customization requirements, and built an edge-cloud collaborative scheduling model, which guaranteed real-time distributed scheduling at the edge. Wang *et al*. [22], aimed at the problem that the large amount of sensor data collected in the industrial sensor cloud system (SCS) was untrustworthy, proposed a data collection and cleaning method based on mobile edge nodes, which can maintained the reliability and integrity of the data. At the same time, it improved the efficiency of data cleaning and greatly reduced the bandwidth and energy consumption of industrial SCS. To address the problem that typical energy-saving and battery life technologies in industrial devices cannot supported dynamic wireless channels in edge computing, Sodhro *et al*. [23] proposed a data reliability model of IoT devices based on edge AI to improve the range and computing speed of IoT devices in industry. Fu *et al*. [24] designed a framework that integrated fog computing and cloud computing to improve the efficiency and security of data storage and retrieval in IIoT in response to data processing, secure data storage, efficient data retrieval, and dynamic data collection in IIoT. Wang *et al*. [25] proposed a cloud-edge computing environment and a CNN-based element segmentation method to solve the problem of visual sorting in customization driven manufacturing systems. The prototype system showed that the proposed method can provided high classification accuracy within an acceptable time. Previous research has been conducted on edge computing in IIoT. The above results provided a good reference point for our research, but personalized and customized production products show multi-variety and small-volume characteristics; besides, the production materials in the production system, including personnel, equipment, materials and product design, scheduling management, routes, and other production processes show the characteristic of continuous dynamic flexibility. Prior research has not considered the new features of personalized customized production, and thus their results do not apply to personalized customized production systems.

Chen *et al.* [26] proposed a decision method based on game theory to realize distributed and efficient computing unloading among device users aimed at the multi-user computing offloading in a multi-channel wireless environment scenario. This method can achieved higher computational offload performance and scale under the condition of increasing user size. Deng *et al.* [27] proposed a dynamic adaptive sequential offloading decision method aimed at the problem that the limited computing resources of the edge computing server of the community and the serious interference between communities limit the scalability of offloading. This method can achieved efficient performance in terms of time delay and energy consumption. Liang *et al.* [28] designed a multi-user collaborative offloading scheduling algorithm based on a decomposition method to solve the coordination problem of wireless resources and computing resources allocation in a multi-user mobile edge computing system under I/O interference, which can made the offloading controllable and has better results. Tran *et al.* [29] proposed a new heuristic algorithm for collaborative task offloading and resource allocation in edge computing, which significantly improved the user's utility for traditional methods. Ruan *et al.* [30] used Lyapunov theory and the proposed deviation update decision algorithm to solve the computing offloading decision and the formulation of the offloading update sequence for the resource allocation problem in the application of the fog computing technology to the sharing mode. This strategy can saved system consumption and improved the resource demand satisfaction rate. In summary, none of these works considered how to choose an edge collaborative model in a customized production environment, and the above-presented decision methods are not suitable for a customized production environment.

Faced with the challenges of stricter task execution time and full utilization of edge device computing resources in personalized customized production, the traditional centralized cloud computing models cannot meet the demand. To build and implement a flexible and reliable edge collaborative computing system, we propose an edge collaborative computing system architecture for personalized customized production and a thing-edge-cloud collaborative computing decision (TCCD) method to achieve efficient computing of customized production tasks and optimal utilization of equipment computing resources. The main contributions of our work can be summarized in three aspects.

1) From the perspective of personalized customized production order task operation, an edge collaborative computing system architecture for personalized customized production is designed for the industrial environment, which helps implement the TCCD method.

2) The TCCD method is analyzed to dynamically divide the customized orders, and a task priority sorting algorithm is proposed to optimize the waiting time of all tasks. Moreover, a decision-making method based on a discrete particle swarm algorithm is proposed to optimize the average execution time of all the tasks and equipment utilization.

3) The proposed TCCD method and traditional methods are compared. To verify its feasibility and effectiveness, the method proposed in this article is implemented on a customized production prototype platform.

The remainder of this article is organized as follows: Section II presents the personalized customized production system architecture used to implement the TCCD method. Section III describes the TCCD method in personalized customization production, explains dynamically dividing personalized customization orders, and proposes a task priority

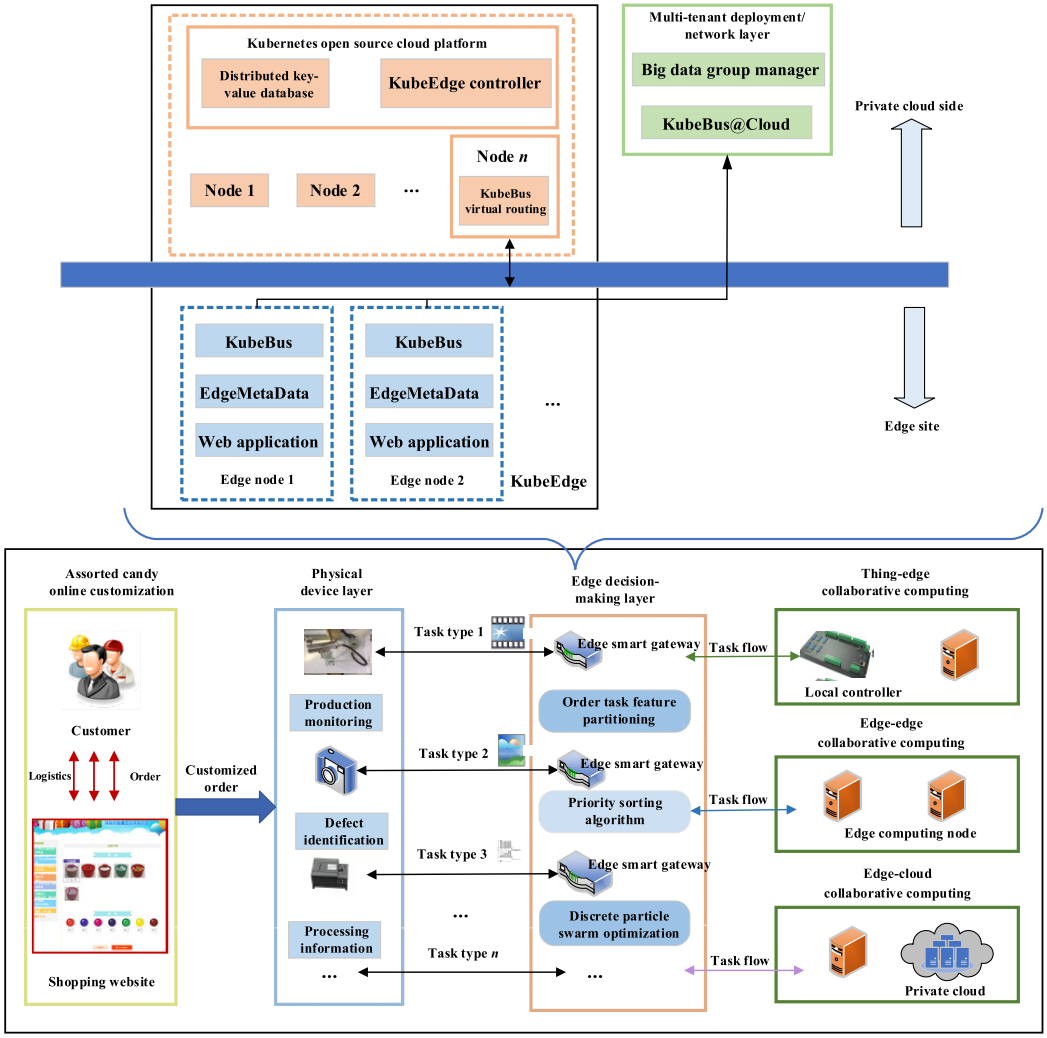

FIGURE 1. The KubeEdge-based edge collaborative computing architecture in a customized production environment.

sorting algorithm and a decision-making method based on a discrete particle swarm algorithm. Section IV experimentally verifies the effectiveness of the proposed TCCD method in personalized customized production and analyzes the experimental results. Section V concludes this article.

II. SYSTEM ARCHITECTURE

To realize the collaborative computing environment of production equipment, edge computing node, and private cloud in the customized production system, a KubeEdge-based edge collaborative computing system architecture is constructed as shown in Fig. 1. KubeEdge builds a homogeneous execution environment for cloud computing and edge computing and connects each edge node, cloud virtual machine, and network container as a VPN. The core of the KubeEdge architecture includes EdgeMetadata service and KubeBus. Edge collaborative computing based on KubeEdge can realize network communication and collaborative computing between production equipment, edge nodes, and the private cloud platform. The EdgeMetadata service is responsible for data storage and synchronization computing when the connection between the edge node and the private cloud platform is unstable. KubeBus provides a software interface for the data communication link between the edge node and the private cloud platform.

The KubeEdge provides a multi-tenant edge infrastructure. On the private cloud data center side, it includes a multi-tenant management/data plane and has a cluster of multiple tenants. The multi-tenant management/data plane includes KubeBus in the cloud center and tenant management functions. A tenant cluster includes one or more edge nodes running in the edge area and a Kubernetes cluster running in the cloud. KubeEdge is deployed in the edge intelligent gateway and realizes the edge collaborative computing function through proxy services. In the private cloud, for each tenant's Kubernetes cluster, a KubeBus virtual router runs on a VM node to route traffic between the edge node subnet and the VM subnet/container network subnet.

Besides, in a private cloud, the server is mainly responsible for controlling edge nodes; receiving information uploaded by edge nodes; assigning tasks; providing IoT applications, such as device management; intelligent production applications, such as virtual factories; and intelligent service applications based on big data analysis. Operators are provided

with a visual interface, and the supervision and scheduling of on-site resources are realized through collaborative computing with edge nodes. In the edge decision-making layer, industrial field resources are connected to the edge intelligent gateway device through the field network. The algorithm model, event management, message routing, and other services deployed in the edge intelligent gateway device perform real-time processing and analysis of the accessed data, on-site reasoning decision-making, conversion, transmission, and so on. Furthermore, the physical device layer, which includes the autonomous guided vehicle (AGV), sensor, and robot, is not only the executing part of the system but also the information acquisition part, and it is mainly responsible for completing the sensing and control work.

Under the framework of an edge collaborative computing system based on KubeEdge technology, this article focuses on the research of the TCCD method for personalized customized production. For the proposed edge collaborative computing system, all the computing tasks are created on on-site devices, including production machines, wireless network nodes, and mobile devices. Tasks are random events and should usually be processed in real-time. For customized production tasks, there are three factors to consider: the amount of computation, the amount of transmission, and the amount of data returned that results from the computation. According to these three factors, the queued waiting time of tasks in a certain time window can be optimized. Then, in the edge collaborative computing system, intelligent tasks can choose different edge collaborative computing methods according to the decision algorithm.

III. THING-EDGE-CLOUD COLLABORATIVE COMPUTING DECISION-MAKING METHOD

Based on the proposed edge collaborative computing architecture, this section analyzes the TCCD method from the perspective of order task division and different edge collaborative computing methods. Moreover, it focuses on optimizing the task queue waiting time, average task execution time, and equipment utilization.

A. REPRESENTATIVE ORDER TASK DIVISION FOR PERSONALIZED CUSTOMIZED PRODUCTION

In the customized production environment, the edge intelligent gateway dynamically divides the customer's customized orders according to the number and types of products in the customer's order received from the private cloud platform. The customized order can be expressed as $[n1\ n2\ n3\ \ldots]$, where $n1$ represents the quantity of product Category 1, $n2$ represents the quantity of product Category 2, and so on. The number of products required by the customer is set to M, the type of products required by the customer is set to F, the number and upper limit of all categories are set to N, and the product category that can be produced by personalized customization is set to G. The steps of dynamic division can be described as follows:

1) When the order is submitted, it is divided into the following three categories according to the quantity of the products required by the customer. When $1 <= M < N/3$, it is set to Class A; when $N/3 <= M < 2N/3$, it is set to Class B; and when $2N/3 < M <= N$, it is set to Class C.

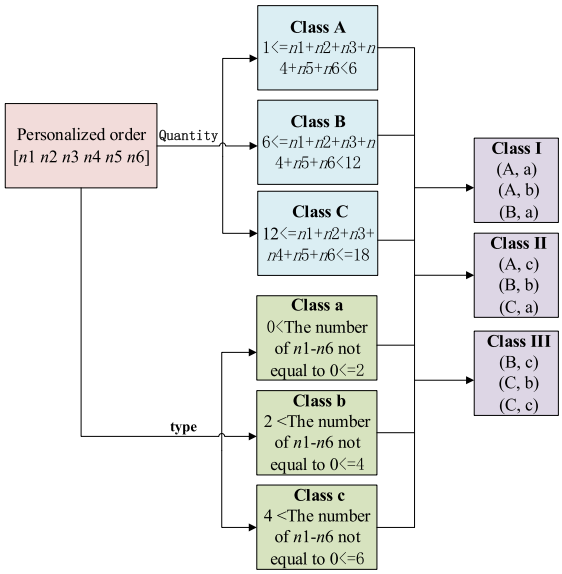

FIGURE 2. Diagram of representative order task division for customized production.

2) When the order is submitted, it is divided into the following three categories according to the types of products required by the customer. When $0 < F <= G/3$, it is set to Category a; when $G/3 < F <= 2G/3$, it is set to Category b; and when $2G/3 < F <= G$, it is set to Category c.

3) Based on the above two classification characteristics, (A, a), (A, b), and (B, a) are classified as Class I, which can be selected for thing-edge collaborative computing, edge-edge collaborative computing, and edge-cloud collaborative computing; (A, c), (B, b), and (C, a) are classified as Class II, which can be selected for edge-edge collaborative computing or edge-cloud collaborative computing; and (B, c), (C, b), and (C, c) are classified as Class III, which can be selected for edge-cloud collaborative computing.

The diagram of the representative order task division for customized production is shown in Fig. 2. The "thing" in thing-edge collaborative computing refers to end-of-things devices, which can also be described as IoT devices, mainly including sensor devices, cameras, and factory machinery equipment. The "edge" in edge-edge collaborative computing and edge-cloud collaborative computing refers to edge computing nodes, which mainly include Raspberry Pi, gateways, routers, and servers. "Cloud" refers to a private cloud platform, mainly a computer cluster with powerful computing, storage, and analysis capabilities.

B. THING-EDGE-CLOUD COLLABORATIVE COMPUTING MODE

The thing-edge collaboration is mainly located at the bottom of the production site, which can integrate the computing resources on the link between production equipment and edge computing nodes to make it fully utilized, capitalize

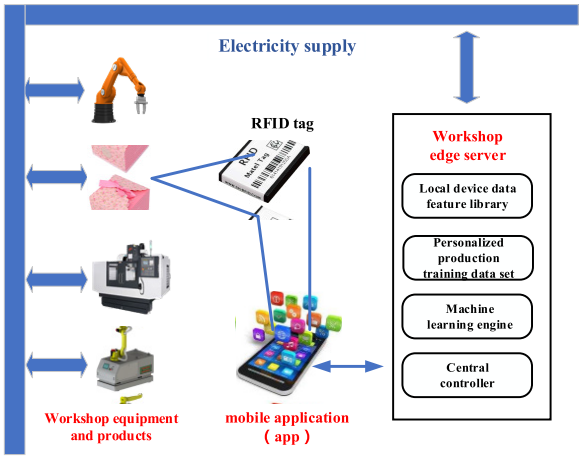

FIGURE 3. The thing-edge collaborative computing mode.

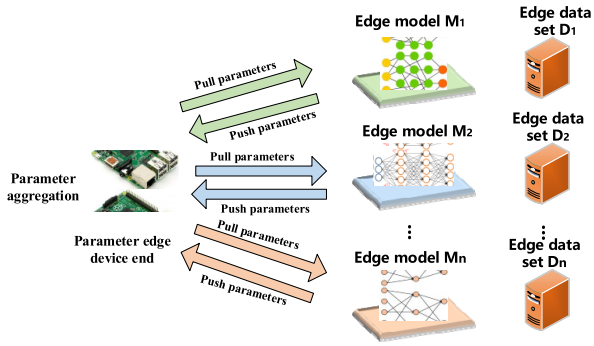

FIGURE 4. The edge-edge collaborative computing mode.

on the advantages of different equipment, and better enhance the capability of edge nodes. The thing-edge collaboration computing mode is shown in Fig. 3. It is widely used in the IoT, especially in smart homes and smart manufacturing. Under the thing-edge collaborative computing mode, the thing is responsible for collecting data and sending it to the edge. Meanwhile, the calculation and control instructions of the edge are received for specific production operations. The edge is responsible for the centralized calculation of multi-channel data, issues instructions, and provides network, computing, and storage services. The thing can perform some simple calculations, and the edge, as the main body of the computing task and the core hub of the system, needs to undertake more computing tasks.

Collaborative computing between the edge-edge infrastructures is a current research hotspot, which can solve the contradiction between the resource requirements of intelligent algorithms and the limited resources and intelligent task requirements of edge devices and the single capability of edge devices. The edge-edge collaboration computing mode is shown in Fig. 4. Specifically, the computing power of a single edge computing node is limited, and, to improve the overall computing power of the system, time-sharing coordination between multiple edge computing nodes is required.

For example, when completing the training task of the deep neural network model, it is not feasible to train in a

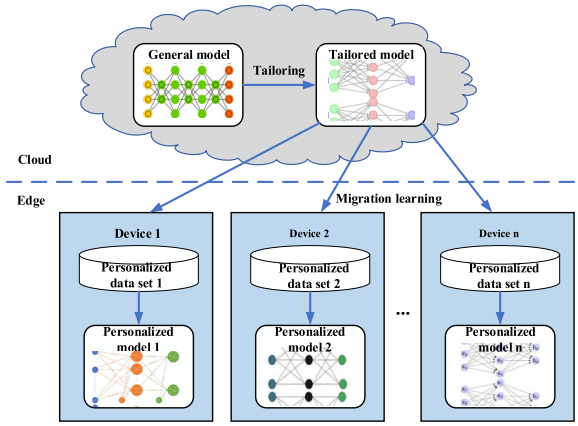

FIGURE 5. The edge-cloud collaborative computing mode.

single edge computing node, which not only consumes a lot of time and computing power but is also easy to overfit the model due to the limitation of data volume to fail to obtain the optimal solution predicted by the model. Therefore, multiple edge compute nodes are required to train the model together. The second is to solve the "data island" problem "data island" in production and manufacturing. The data source of a certain edge computing node has a strong locality and needs to cooperate with other edge computing nodes to complete a larger range of tasks. For example, in the operation monitoring of the customized production line of the whole factory, generally, one edge computing node can only obtain the operating status information of one workshop, and the cooperation among multiple edge computing nodes can be combined into an overview diagram of the workshop operating status of the whole intelligent factory.

In edge-cloud collaborative computing, the edge is responsible for data computing and storage in the local area, and the cloud is responsible for big data analysis, mining, and algorithm training optimization. The edge-cloud collaboration computing mode is shown in Fig. 5. The collaboration of the edge-cloud can be divided into two parts. The first is functional collaboration. This kind of collaboration assumes different functions based on different geographic spaces and roles of different computing devices. For example, the edge is responsible for preprocessing, and the cloud is responsible for multi-channel data processing and service provision. The second is performance collaboration. This is due to the limitation of computing power, and computing devices of different levels undertake tasks with different computing power requirements, including longitudinal cutting and assignment of tasks.

C. THING-EDGE-CLOUD COLLABORATIVE COMPUTING DECISION-MAKING ALGORITHM

In the thing-edge-cloud collaborative computing system, $C_i = \{J^c, B^c, Q^c\}$ represents cloud node C_i, where J^c represents the computing capacity of cloud node i, B^c represents the available bandwidth (in Mbps) between the cloud node and smart devices, and Q^c represents the cost of processing

TABLE 1. Notations

Notations	Definition
C_i	Cloud node.
E_i	Edge node.
S_i	Smart device.
B	Network bandwidth.
J	Computing capacity.
Z_t	Equipment utilization.
T_t	The execution time of the task.
I_j	Decision variable.
X_{ij}	The j-th dimension of the current position of i-th particle.
V_{ij}	The j-th dimension of the current velocity of i-th particle.
$PBest_{ij}$	The j-th dimension of the optimal position in the history state of i-th particle.
$GBbest_{ij}$	The j-th dimension of the historical optimal position of all particles in the entire particle swarm.
k	Iterative algebra.
r_1, r_2	Two random numbers uniformly distributed between [0, 1].
c_1, c_2	Learning factor (normal number, which is generally taken as 2).
w	Inertia weight.

unit data on the cloud node. There are many edge computing nodes around smart devices to provide edge computing services. Moreover, $E_i = \{J^e, B^e, Q^e\}$ represents edge node E_i, where J^e represents the computing capacity of cloud node i, B^e represents the available bandwidth (in Mbps) between the edge node and smart devices, and Q^e represents the cost of processing unit data on the edge node. Smart device refers to any kind of equipment, apparatus or machine with computational processing capabilities. S_i refers to a certain smart device, whose computing power is represented by J^s, and the task generated by the smart device can be represented as $T_j = \{U_j, D_j, R_j\}$, where U_j is the input data size uploaded by task j, D_j is the computational volume of the task processed on the cloud or edge node, and R_j is the data size returned by the computational results of the task. When choosing a collaborative computing mode for tasks, there are two main optimization goals for a customized production system, including increasing the utilization of edge computing nodes and reducing the average execution time of all tasks. The main notations description used in the thing-edge-cloud collaborative computing decision-making are given in Table 1.

Task execution time is not only a factor of great concern in the personalized customization production process but also an important indicator of personalized customization production efficiency. Task execution time can be divided into four parts in the process of selecting edge collaborative computing mode.

1) The selection of decision stage t_d mainly includes information collection and making decisions based on the collected information and the necessary waiting time.

2) In the data transmission stage t_t, after making a decision, data needs to be transmitted from the source (thing) node to the edge computing node or cloud node. The transmission time can be expressed as:

$$t_t = U/B \tag{1}$$

where U is the upload data size of the task, and B (B^c or B^e) is the network bandwidth.

3) The task execution stage t_e refers to the time to execute tasks on smart devices, edge nodes, or cloud nodes. The time cost is as follows:

$$t_e = D/J \tag{2}$$

where D is the amount of task data that needs to be processed on smart devices, edge nodes, or cloud nodes; and J (J^s, J^e or J^c) is the computing capacity of smart devices, edge nodes, or cloud nodes.

4) The result return stage t_r refers to the time required to return the result data from the edge node or cloud node to the smart device. Here, we assume that the bandwidth of the two-way communication is stable and consistent. The time can be expressed as:

$$t_r = R/B \tag{3}$$

where R is the data size of the task result, and B (B^c or B^e) is the network bandwidth.

Then, the execution time of task decision for collaborative computing is:

$$T_t = t_d + (U + R)/B + D/J \tag{4}$$

In general, the purpose of selecting the task computing method is to reduce the task execution time, that is, the execution time meets the following conditions:

$$T_s > t_d + (U + R)/B + D/J \tag{5}$$

where T_s is the time to execute the task on the smart device, and the other parameters are the same as in Equation (4). If thing-edge collaboration, edge-edge collaboration, or edge-cloud collaborative computing perform tasks faster than smart devices, then it is worthwhile to choose the edge collaborative computing mode.

Furthermore, a task priority sorting algorithm is proposed to optimize the waiting time of tasks in the queue. The task priority sorting algorithm is shown in Algorithm 1. Tasks are being prioritized based on the previous results. First, according to the task characteristic model established above, the computing amount, transmission amount, and amount of data returned by the computing result of each task are obtained. Then, the priority of the computing amount, transmission amount, and the amount of data returned from the computing result is set to 0 or 1 because the amount of data returned from the computing result is usually relatively small, which is ignored. Finally, if the priority of the computing and transmission of the task are both 1, then the task is a high priority task; if the priority of the computing and transmission of the task are both 0, then the task is a low priority task; otherwise, the task is a medium priority task.

Algorithm 1 Pseudocode of Task Priority Sorting Algorithm

```
Initialization: Input task number N^t, task T_j, computing amount of task D, transmission amount of task U
Output: sorted task flow L
Begin
    Initialize task flow randomization
    Set the priority of D, U to 0 or 1
    for j = 1; j <= N^t; j++
        for i = 0; i < N^t; i++
            if the priority of D and U are 1 then
                P(T_j) = 0 // T_j is a high priority task
                L[i] = T_j
                if the priority of D and U are 0 then
                    P(T_j) = 1 // T_j is a low priority task
                    L[i] = T_j
                else
                    P(T_j) = 1/2 // T_j is a medium priority task
                    L[i] = T_j
                end if
            end if
        end for
    end for
    Reorder the elements in L using the insertion sort algorithm
    return L
End
```

Equipment utilization is another important indicator that must be considered when making thing-edge-cloud collaborative computing decisions. In customized production, the improvement of equipment utilization can make full use of the computing resources of the equipment and improve the flexibility and intelligence of the production line. Equipment utilization refers to the ratio of the time taken to perform tasks on edge nodes to the time taken to complete tasks, which can be expressed as:

$$Z_t = D_e \big/ (J * T_t) \tag{6}$$

where Z_t represents equipment utilization, D_e represents the amount of task data to be processed on the edge node, and T_t represents the execution time of the task.

Based on the above model, the task execution time and equipment utilization are often related to the edge collaborative computing method of task selection, and thus this article defines a decision variable $I_j \in \{-1, 0, 1\}$ to represent the edge collaborative computing method of a task, and this variable is also the decision amount of selection. This variable can be expressed as follows:

$$I_j = \begin{cases} -1, & \text{if task } j \text{ is TEC computing} \\ 0, & \text{if task } j \text{ is EEC computing} \\ 1, & \text{if task } j \text{ is ECC computing} \end{cases} \tag{7}$$

where, $I_j = -1$ if task j is the thing-edge collaborative (TEC) computing mode;$I_j = 0$ if it is the edge-edge collaborative (EEC) computing mode; and, otherwise, the task is the edge-cloud collaborative (ECC) computing mode, and then $I_j = 1$.

Next, we discuss the TCCD algorithms for optimizing execution time and device utilization. Choosing the TCCD method for tasks with a large amount of computing in personalized customization production should simultaneously reduce task execution time and improve equipment utilization. The optimization objectives are as follows:

$$\min \left[\left(\alpha * \overline{T_t} + \beta * 1 \big/ \overline{Z_t} + \gamma * N^t * Q \right) * I \right] \tag{8}$$

where $\overline{T_t}$ is the normalized value of the execution time of the task; $\overline{Z_t}$ is the normalized value of equipment utilization; N^t is the number of tasks to be processed on the edge node or cloud node; Q (Q^e or Q^c) is the cost of processing unit data on the edge node or cloud node; $N^t * Q$ represents the cost of TCCD task computing mode; α, β, and γ are the weights of execution time, equipment utilization, and cost of edge collaborative computing, and $\alpha + \beta + \gamma = 1$. These parameters can help the customized production system to adjust the bias of task execution time, equipment utilization, and cost. Besides, I represents the decision variable of the edge collaborative computing mode selection, that is, the edge collaborative computing mode selected for each task (whether to choose the TEC computing, the EEC computing, or the ECC computing mode. It satisfies $I = [I_1, I_2, \ldots, I_j, \ldots, I_{N^t}]$ and $I_j \in \{-1, 0, 1\}$). The other parameters have the same meaning as before.

Since there are three options for edge collaborative computing for each task, there are $3N^t$ possible solutions for TCCD for N^t tasks. Therefore, this is a nonlinear restricted programming problem, and the problem cannot be solved by a formula. If the enumeration method is used to traverse the entire solution set to find the optimal solution, then the computational time complexity is very high, and thus this method is not suitable for solving this problem. Therefore, it is extremely necessary to design a fast and low-time complexity heuristic algorithm to find the optimal solution and reduce the computational complexity. It can be observed that the TCCD variable I is a vector with only discrete values. This study used the discrete particle swarm algorithm to solve this nonlinearly restricted planning problem.

Based on this, the standard particle swarm optimization (PSO) is only suitable for searching for optimal solutions in a continuous space, and it cannot be used directly for discrete spaces.

In the traditional PSO algorithm [31], all particles have their own positions and speeds. The meaning of particle position is a feasible solution in the solution space, while the particle speed represents the distance between the next position of a particle and the current position.

In the solution space of the J-dimensional optimization problem, the update formulas for the position and velocity of the j-th dimension of i-th particle are as follows:

$$V_{ij}^{k+1} = wV_{ij}^k + c_1 r_1 (PBest_{ij}^k - X_{ij}^k) + c_2 r_2 (GBest_j^k - X_{ij}^k) \tag{9}$$

and

$$X_{ij}^{k+1} = X_{ij}^{k} + V_{ij}^{k+1} \tag{10}$$

To solve the problem of minimizing the task execution time and maximizing equipment utilization rate of intelligent equipment in the personalized customized production environment constructed in this article, a discrete PSO (DPSO) algorithm based on the PSO algorithm is adopted.

In DPSO, each dimension X_{ij} of the particle position and each dimension $PBest_{ij}$ of the optimal historical position of the particle are set to discrete values -1, 0, or 1, and the particle velocity V_{ij} is the same as in the PSO algorithm. The function is defined as follows, where the parameter is the j-th dimension of the current velocity of i-th particle:

$$S(V_{ij}^{k+1}) = \left[\left(V_{ij}^{k+1}\right)^2 - 1\right] \Big/ \left[\left(V_{ij}^{k+1}\right)^2 + V_{ij}^{k+1} + 1\right] \tag{11}$$

As shown, this function is a monotonically increasing function with a value range of $(-1, 1)$. The function value of the particle's velocity can be regarded as the probability that the particle's position is -1, 0, or 1, which is consistent with the decision variables of the edge computing of personalized customized production. When the velocity value is large, the probability of particle position going to 1 is greater; when the velocity value is small, the probability of particle position going to -1 is greater; when the velocity value is small, the probability of particle position going to 0 is greater. Then, the j-th dimension of the current position of i-th particle in the DPSO algorithm is updated to the following formula:

$$X_{ij}^{k+1} = \begin{cases} 1, & \text{if } R_{ij} < S(V_{ij}^{k+1}) \\ -1, & \text{if } R_{ij} > S(V_{ij}^{k+1}) \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

where R_{ij} is a random number uniformly distributed in the interval $[-1,1]$.

In this article, the position of the particle is the decision variable of the execution position of each task for the optimization of the execution time and equipment utilization of the TCCD method for personalized production. Using the function of Equation (12), the speed can be used as a parameter to choose whether the position of the particle is $-1, 0$, or 1. According to DPSO, the first step is initialization, that is, to randomly assign decision variables to the particle swarm and then calculate the corresponding execution time of each particle and the equipment utilization value at this time through Equations (4) and (6), and the historical optimal position of each particle and the historical optimal position of the population are obtained. The next step is a cyclic process in which the global optimal value of the problem is approached continuously. For each particle, the following operations are performed: update the particle velocity and position, calculate the equipment utilization rate and order task execution time corresponding to the new decision variable, and update the historical optimal position of each particle and the historical optimal position of the population according to the least principle of (8). When the set number of cycles is reached, a relatively optimal decision variable can be obtained to minimize the average execution time of all the order tasks and maximize equipment utilization. Therefore, the specific description of the TCCD method algorithm based on discrete PSO is shown in Algorithm 2. For this nonlinear constrained programming problem, the time complexity of DPSO is polynomial, while the time complexity of the exhaustive method is exponential. As shown, the edge cooperative decision algorithm based on discrete PSO has the advantages of low time complexity and reduced computational complexity.

Algorithm 2 Pseudocode of the TCCD method based on discrete particle swarm algorithm

```
Initialization: Input L obtained by Algorithm 1, task
T_j//T_j belongs to L
Output: optimal task decision variable I, minimum task
execution time T_t, maximum equipment utilization Z_t
Begin
    for each particle T_j
        Initialize velocity V_ij and position X_ij for particle
        T_j
        Calculate PBest_ij and GBest
        Evaluate particle T_j and set PBest_ij = X_ij
    end for
    GBest = min{PBest_ij} //I = GBest
    while did not reach the iteration termination number
    k do
        for i = 1 to L.size() // L.size() is the scale of the
        particle
            Update the velocity and position of particle
            T_j
            Calculate T_t and Z_t and get F(T_j) according
            to (8)
            Evaluate particle T_j
            if F(T_j) > min(F) then
                Give up the current position of particle T_j
            else
                if fit(X_ij) < fit(PBest_ij) then
                    PBest_ij = X_ij
                end if
                if fit(PBest_ij) < fit(GBest) then
                    GBest = PBest_ij
                end if
            end if
        end for
    end while
    return I, T_t, Z_t
End
```

IV. EXPERIMENTS AND ANALYSIS

In this section, we describe an actual customized production prototype platform as a case study to verify the effectiveness of the proposed TCCD method.

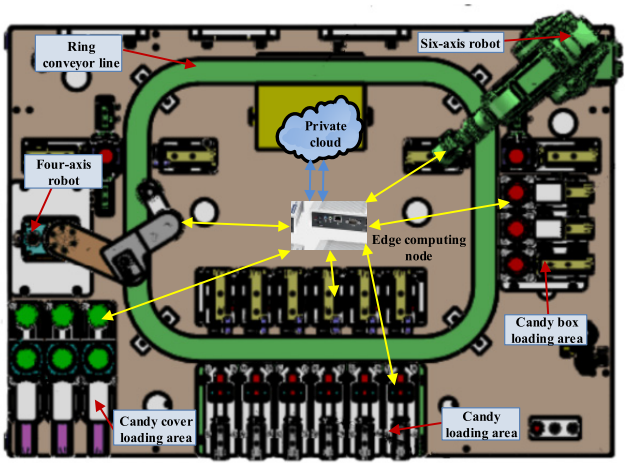

FIGURE 6. Prototype platform used for the TCCD method in the customized production of candy packaging.

A. PROTOTYPING PLATFORM AND EXPERIMENTAL SETUP

This case study is based on an actual production environment of a multi-variety and small-batch candy packaging intelligent production prototype platform. The layout of the prototype platform production line is shown in Fig. 6, which consisted of three parts: the physical device layer, the edge computing layer, and the industrial private cloud server layer. The physical device layer included multiple types of devices, such as manipulators, AGVs, photoelectric sensors, and RFID. In the edge computing layer, Raspberry Pi and edge intelligent gateway had typical application features of edge computing nodes that provided data processing, computation, status monitoring, and control functions for field devices in the prototype platform. Moreover, all the device data was connected to the platform's private cloud computing center via industrial networks. Through the private cloud computing center, instructions for allocating various real-time tasks and monitoring equipment status were generated, and personalized customized production was managed in real-time through edge computing. At the same time, wired Ethernet and wireless communication systems were used for information interaction between different layers of the prototype platform. All the edge computing nodes were connected to the private cloud center through KubeEdge. Real-time tasks and events were executed by the edge computing node unit and perform related computations. Besides, long-term data was sent to the private cloud center for storage in accordance with the instruction cycle for subsequent offline analysis.

Then, based on the prototype platform, performance verification experiments were conducted. The prototype platform was a candy packaging production line. The basic process was as follows: First, customers selected their favorite candy on the app or web page to buy it online, and then the order information was sent to the manufacturing prototype platform. After that, intelligent agents with edge collaborative computing capabilities completed production tasks in a self-organizing manner. Finally, the completed customized orders were automatically transported to the warehouse by

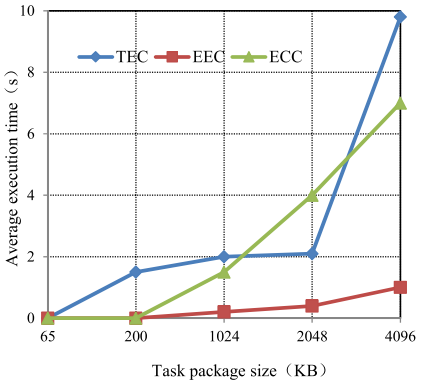

FIGURE 7. Average execution time of the collaborative computing modes.

the logistics system of the customized production factory. In this experiment, a neural network model with a size of 65 KB-4 MB was executed on the edge computing node, and the number and type of candies in a customized candy packaging order were used as the input of the neural network model, which can be used to predict the completion time of the personalized order. The network bandwidth supported by each edge node and smart device is 100Mbps. The computing capacity of smart devices and edge nodes are 500 MHz, 1.4 GHz respectively. The CPU processing frequency of the cloud server is 8GHz, and the transmission rate is 5Mbps. To ensure the reliability of the experimental results, for each neural network model of different sizes, we carried out repeated experiments.

At the same time, the average execution time and the equipment utilization were used as evaluation indicators to evaluate the performance of the TCCD method for customized production. First, we compared the average execution time of task packages based on the KubeEdge framework's TEC, EEC, and ECC computing modes. We compared the performance of the proposed TCCD method with the three traditional methods. Traditional comparison methods included a random decision-making method (RAND) (assuming that each collaborative computing method had the same probability of being selected, the task randomly selected the collaborative computing method), the minimum execution time decision method (METD) (dynamic calculation of the capabilities of each computing node, and edge collaborative computing with smaller execution time was selected), and a deviation update decision-making method based on Lyapunov theory (LDUD) [30]. In the decision-making process of edge collaborative computing, we set $\alpha = 0.4$, $\beta = 0.4$, and $\gamma = 0.2$. Finally, the influence of α, β, and γ on the experimental results were analyzed.

B. RESULT AND ANALYSIS

The relationship between the KubeEdge based framework and the average execution time under the change of task package size is presented in Fig. 7, which can verify the advantages of the KubeEdge framework-based proposed in this article. As shown, the average execution time of the collaborative computing modes increased exponentially with the

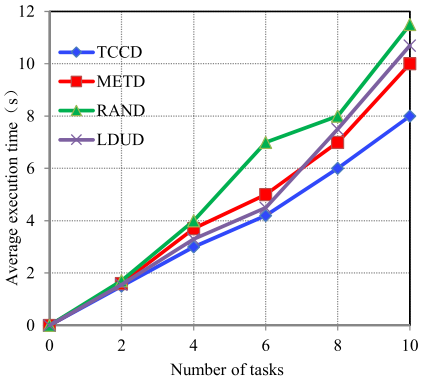

FIGURE 8. Average execution time under different numbers of tasks.

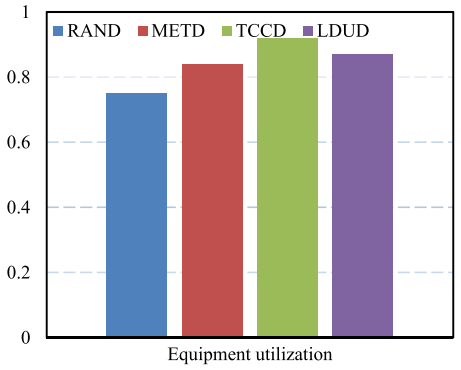

FIGURE 9. Equipment utilization under different decision-making methods.

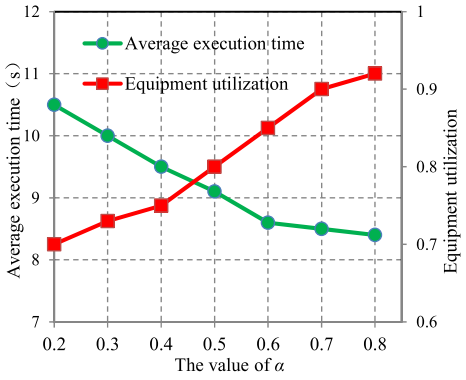

FIGURE 10. Weight parameter analysis.

increase of the task package size. Furthermore, the EEC computing mode was faster than the TEC and ECC computing modes, especially for larger task packages (i.e., greater than 200 KB). The average execution time was less than 1 second for the task packet that was less than 200 KB, while the peer-to-peer communication between edge computing nodes still had a low delay. Moreover, due to the low computing capability of the physical device, when the task package was larger than 2 MB, it was sent step by step, and the average execution time increased rapidly. We observed that it was not feasible to exchange task packages larger than 32 MB in the TEC and EEC calculation methods. This is probably due to the hardware and memory limitations of the embedded development board. Therefore, to reduce the number of messages transmitted to the private cloud, this article used KubeEdge as an edge orchestration device cluster structure to achieve the effect of reducing costs and increasing the number of devices that can support the specific available bandwidth.

The effect of the number of tasks on the average execution time is presented in Fig. 8. As shown, with the increased number of tasks, the average execution time of each decision mechanism increased, and the proposed TCCD method showed obvious advantages. When the number of tasks was 10, the average execution time of the TCCD method was 30%, 20%, and 25% lower than that of the RAND, METD, and LDUD methods, respectively. This is because the RAND method randomly selected the edge collaborative computing method for tasks, which was a relatively blind method. Sometimes it took a long time to wait, and thus the effect was not good. The METD method was more scientific in choosing edge collaborative computing, but the process was complicated and time-consuming, and thus the result was not the best. The LDUD method focused more on the stability of the system, and its effect was even worse than METD method when the number of tasks was large. The proposed TCCD method considered a variety of factors and placed particular emphasis on the cost of various times, which resulted in relatively ideal results.

The equipment utilization of different methods with a certain number of tasks is presented in Fig. 9. The results showed that, for the four decision strategies, the TCCD method had the highest equipment utilization rate, followed by the LDUD and METD methods, while the RAND method was the worst. The RAND method had relatively large randomness and unstable performance, and thus the equipment utilization rate of the RAND method was the lowest. The METD method adopted the strategy of the least time principle, and better results can be obtained. However, this method did not consider the equipment utilization factor in the decision strategy, and thus its performance was not optimal. The LDUD method calculated the deviation value according to the optimal decision results of all tasks obtained in the previous time to select the collaborative calculation method of tasks, and its performance was better. However, this method has the disadvantage of high computational complexity and time complexity. The TCCD method proposed in this article comprehensively considered the two factors of time and equipment utilization, which has small time complexity and can reduce the computational complexity, and its performance was relatively best.

To analyze the influence of the parameters α and β on the experimental results, we set $\gamma = 0.2$ and changed the values of α and β. The trend of the average execution time and equipment utilization with α and β is presented in Fig. 10. The average execution time decreased with the increase of α value. This means that a greater value of α means there is a greater weight of the execution time in the TCCD and the system will choose the edge collaborative computing method with a shorter delay. Similarly, a larger value of β means the system will choose the edge collaborative computing method

with higher equipment utilization. In summary, the parameters α, β, and γ are the weights of execution time, equipment utilization, and cost, respectively. Producers can adjust different parameter values according to their own preferences to meet the needs of specific scenarios.

C. DISCUSSION

In the previous part, we presented the results of the average execution time of the task package based on the KubeEdge framework, the performance of the TCCD method, and the weight parameter analysis. Compared with TEC and ECC, EEC had a lower average execution time, which is due to the efficient message encapsulation ability of the KubeEdge framework, which can greatly reduce the communication pressure. When the four decision-making methods are compared, the TCCD method proposed in this article can usually produce the best results in some situations. However, within the permissible range, the system may have some decision errors, which may reduce the quality of service. In terms of decision costs, considering more factors may be more costly than considering fewer factors. Moreover, regarding the decision-making strategy, the RAND method is a simple method, but the cost is a high average task execution time and low equipment utilization.

Although the METD method has advantages in the average execution time, in the actual customized production process, not only the production efficiency was considered but also the quality and cost of the production were also guaranteed. Although the LDUD method has advantages in equipment utilization, it is inferior in average execution time. Besides, in the experiment, we only selected a lighter weight model as the experimental load, and the number was small, which may only meet the needs of small-scale customized production. Determining the combined values of the parameters α, β, and γ to better serve the edge of the network is a problem worthy of discussion. For example, to avoid downtime in the event of equipment failure, the value of β can be increased to improve equipment utilization for on-time delivery. Therefore, the values of the parameters α, β, and γ mainly depend on actual production requirements. Considering that urgent orders, process changes, and real-time processing of product quality events exist in actual production applications, the collaborative mechanism between edge computing nodes and cloud computing nodes and the real-time processing method of tasks should be further studied to improve the flexibility and efficiency of personalized customization production.

V. CONCLUSION

In the production of personalized customization, edge computing, as the middle layer of the IoT architecture, provides real-time computing, storage, and communication mechanism at the field level. Edge devices with limited memory resources cannot compute tasks that require high computing capability, especially when multiple tasks are running on the edge device at the same time. In this article, the TCCD method for customized production was studied. First, a personalized customized production system architecture for implementing the TCCD method was proposed. Then, a task priority sorting algorithm was proposed to optimize the waiting time of all tasks in the order. A TCCD method based on discrete particle swarm algorithm was proposed to optimize the average execution time of all the order tasks and equipment utilization. Finally, the feasibility of the proposed TCCD method was verified by using a candy packaging personalized customization prototype platform. The experimental results showed that the proposed TCCD method was significantly better than the RAND, METD, and LDUD methods in terms of average task execution time and equipment utilization. In general, the proposed TCCD method for personalized customized production can provide a strong impetus for the unified management, scheduling, operation, and maintenance of infrastructure resources on the production edge.

REFERENCES

[1] B. Chen, J. Wan, Y. Lan, M. Imran, D. Li, and N. Guizani, "Improving cognitive ability of edge intelligent IIoT through machine learning," *IEEE Netw.*, vol. 33, no. 5, pp. 61–67, Sep. 2019.
[2] J. Wan, J. Li, M. Imran, D. Li, and Fazal-e-Amin, "A blockchain-based solution for enhancing security and privacy in smart factory," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3652–3660, Jun. 2019.
[3] X. Hu, J. Wan, T. Wang, and Y. Zhang, "An IoT-based cyber-physical framework for turbine assembly systems," *IEEE Access*, vol. 8, pp. 59732–59740, 2020.
[4] J. Wan, B. Yin, D. Li, A. Celesti, F. Tao, and Q. Hua, "An ontology-based resource reconfiguration method for manufacturing cyber-physical systems," *IEEE/ASME Trans. Mechatronics*, vol. 23, no. 6, pp. 2537–2546, Dec. 2018.
[5] J. Wan, S. Tang, D. Li, M. Imran, C. Zhang, C. Liu, and Z. Pang, "Reconfigurable smart factory for drug packing in healthcare industry 4.0," *IEEE Trans. Ind. Informat.*, vol. 15, no. 1, pp. 507–516, Jan. 2019.
[6] J. Wan, B. Chen, M. Imran, F. Tao, D. Li, C. Liu, and S. Ahmad, "Toward dynamic resources management for IoT-based manufacturing," *IEEE Commun. Mag.*, vol. 56, no. 2, pp. 52–59, Feb. 2018.
[7] H. Liao, Z. Zhou, X. Zhao, L. Zhang, S. Mumtaz, A. Jolfaei, S. H. Ahmed, and A. K. Bashir, "Learning-based context-aware resource allocation for edge-computing-empowered industrial IoT," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4260–4277, May 2020.
[8] M. S. Jawad, M. Bezbradica, M. Crane, and M. K. Alijel, "AI cloud-based smart manufacturing and 3D printing techniques for future in-house production," in *Proc. Int. Conf. Artif. Intell. Adv. Manuf. (AIAM)*, Oct. 2019, pp. 747–749.
[9] S. Guo, T.-M. Choi, B. Shen, and S. Jung, "Inventory management in mass customization operations: A review," *IEEE Trans. Eng. Manag.*, vol. 66, no. 3, pp. 412–428, Aug. 2019.
[10] F. Tao, J. Cheng, and Q. Qi, "IIHub: An industrial Internet-of-Things hub toward smart manufacturing based on cyber-physical system," *IEEE Trans. Ind. Informat.*, vol. 14, no. 5, pp. 2271–2280, May 2018.
[11] Z. Meng, Z. Wu, and J. Gray, "Architecting ubiquitous communication and collaborative-automation-based machine network systems for flexible manufacturing," *IEEE Syst. J.*, vol. 14, no. 1, pp. 113–123, Mar. 2020.
[12] K. Kaur, S. Garg, G. S. Aujla, N. Kumar, J. J. P. C. Rodrigues, and M. Guizani, "Edge computing in the industrial Internet of Things environment: Software-defined-networks-based edge-cloud interplay," *IEEE Commun. Mag.*, vol. 56, no. 2, pp. 44–51, Feb. 2018.
[13] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5031–5044, May 2019.
[14] M. Fazio, R. Ranjan, M. Girolami, J. Taheri, S. Dustdar, and M. Villari, "A note on the convergence of IoT, edge, and cloud computing in smart cities," *IEEE Cloud Comput.*, vol. 5, no. 5, pp. 22–24, Sep. 2018.
[15] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.

[16] Z. Zhou, Y. Guo, Y. He, X. Zhao, and W. M. Bazzi, "Access control and resource allocation for M2M communications in industrial automation," *IEEE Trans. Ind. Informat.*, vol. 15, no. 5, pp. 3093–3103, May 2019.
[17] C.-C. Lin, D.-J. Deng, Y.-L. Chih, and H.-T. Chiu, "Smart manufacturing scheduling with edge computing using multiclass deep q network," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 4276–4284, Jul. 2019.
[18] Y. K. Lee, S.-J. Lee, H. Lee, and D. Yoon, "Implementation of distributed smart factory platform based on edge computing and OPC UA," in *Proc. 45th Annu. Conf. IEEE Ind. Electron. Soc. IECON*, Oct. 2019, pp. 4235–4239.
[19] B. Chen, J. Wan, A. Celesti, D. Li, H. Abbas, and Q. Zhang, "Edge computing in IoT-based manufacturing," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 103–109, Sep. 2018.
[20] J. Wan, J. Yang, S. Wang, D. Li, P. Li, and M. Xia, "Cross-network fusion and scheduling for heterogeneous networks in smart factory," *IEEE Trans. Ind. Informat.*, vol. 16, no. 9, pp. 6059–6068, Sep. 2020.
[21] J. Ma, H. Zhou, C. Liu, M. E, Z. Jiang, and Q. Wang, "Study on edge-cloud collaborative production scheduling based on enterprises with multi-factory," *IEEE Access*, vol. 8, pp. 30069–30080, 2020.
[22] T. Wang, H. Ke, X. Zheng, K. Wang, A. K. Sangaiah, and A. Liu, "Big data cleaning based on mobile edge computing in industrial sensor-cloud," *IEEE Trans. Ind. Informat.*, vol. 16, no. 2, pp. 1321–1329, Feb. 2020.
[23] A. H. Sodhro, S. Pirbhulal, and V. H. C. de Albuquerque, "Artificial intelligence-driven mechanism for edge computing-based industrial applications," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 4235–4243, Jul. 2019.
[24] J.-S. Fu, Y. Liu, H.-C. Chao, B. K. Bhargava, and Z.-J. Zhang, "Secure data storage and searching for industrial IoT by integrating fog computing and cloud computing," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4519–4528, Oct. 2018.
[25] Y. Wang, K. Hong, J. Zou, T. Peng, and H. Yang, "A CNN-based visual sorting system with cloud-edge computing for flexible manufacturing systems," *IEEE Trans. Ind. Informat.*, vol. 16, no. 7, pp. 4726–4735, Jul. 2020.
[26] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.
[27] M. Deng, H. Tian, and X. Lyu, "Adaptive sequential offloading game for multi-cell mobile edge computing," in *Proc. 23rd Int. Conf. Telecommun. (ICT)*, May 2016, pp. 1–5.
[28] Z. Liang, Y. Liu, T.-M. Lok, and K. Huang, "Multiuser computation offloading and downloading for edge computing with virtualization," *IEEE Trans. Wireless Commun.*, vol. 18, no. 9, pp. 4298–4311, Sep. 2019.
[29] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, Jan. 2019.
[30] L. Ruan, Z. Liu, X. Qiu, Z. Wang, S. Guo, and F. Qi, "Resource allocation and distributed uplink offloading mechanism in fog environment," *J. Commun. Netw.*, vol. 20, no. 3, pp. 247–256, Jun. 2018.
[31] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw., ICNN*, Perth, WA, Australia, Nov. 1995, pp. 1942–1948.

CHUN JIANG received the B.Eng. degree in measuring and controlling technology and instrument from the North University of China, Taiyuan, China, in 2015. She is currently pursuing the Ph.D. degree with the South China University of Technology, Guangzhou. Her research interests include edge computing and cyber-physical systems.

JIAFU WAN (Member, IEEE) is currently a Professor with the School of Mechanical & Automotive Engineering, South China University of Technology, China. He has directed 20 research projects, including the National Key Research and Development Program of China, the Key Program of National Natural Science Foundation of China, and the Guangdong Province Key Areas Research and Development Program. Thus far, he has published more than 150 scientific articles, including 100+ SCI-indexed papers, 40+ IEEE Transactions/Journal papers, 20 ESI Highly Cited Papers, and 4 ESI Hot Papers. According to the Google Scholar, his published work has been cited more than 13 000 times (H-index = 50). His other SCI citations, sum of times cited without self-citations reached 3,500 times (H-index = 37) according to the Web of Science Core Collection. He is an Associate Editor of the IEEE/ASME TRANSACTIONS ON MECHATRONICS, the *Journal of Intelligent Manufacturing and Computers & Electrical Engineering*, and Editorial Board of *Computer Integrated Manufacturing Systems*. His research interests include cyber-physical systems, intelligent manufacturing, big data analytics, industry 4.0, smart factory, and cloud robotics. He is listed as a Clarivate Analytics Highly Cited Researcher in 2019 and 2020.