Research on key technologies of edge computing in discrete manufacturing industry in the era of big data

Guanghua Lu *, Mingbo Liu, Zenghai Wang, Lei Gao

State Grid Information and Communication Industry Group Co., Ltd. Beijing Branch
Beijing, 10031China

* LuGuanghua1@sgitg.sgcc.com.cn

Abstract. Relying on the centralized operation feedback mode of cloud computing, the
data processing capacity is improved, but the delay and energy consumption of discrete
manufacturing industry are increased. A resource scheduling method for discrete
manufacturing based on edge computing architecture is proposed. Based on the
establishment of big data multi cluster edge cloud framework, the task priority of
discrete manufacturing industry is determined. According to the task priority of each
production link, the objective plan of resource scheduling and unloading is established
to achieve real-time and efficient task resource scheduling. The experimental results
show that the stability is higher than 78%, the real-time response rate is fast and the
performance is better.

Keywords: big data; discrete manufacturing; edge computing; resource scheduling;
resource offloading.

1. Introduction

The discrete manufacturing industry relies on cloud computing model to process data to realize the
scheduling of resources and tasks in each production link. Compared with the process manufacturing
industry, the discrete manufacturing industry has more dispersed production links, more diverse
production equipment, and easier to transform the process through software. Because of its high level
of automation, mature and closed production links, and the production process and capacity are mainly
determined by hardware, a stable and reliable resource scheduling technology is urgently needed to
solve the above problems of discrete manufacturing enterprises [1].

The traditional manufacturing resource scheduling method based on cloud computing model relies
on the advantage of strong processing capacity of cloud computing to centralize all the data in the
manufacturing industry and save the storage computing space of hardware [2]. However, it also reveals
some disadvantages, such as poor real-time performance, while the traditional cloud computing model
has to transfer data to the cloud computing center before requesting the data processing results, which
increases the delay of the production and processing system. The discrete manufacturing industry edge
computing technology is mainly designed to efficiently and accurately schedule the resources of each
discrete node within the manufacturing industry, optimize each production link in the manufacturing
industry, and improve the overall management and production efficiency of the discrete manufacturing
industry [3]. Based on the above analysis content, this paper will study a discrete manufacturing industry...
resource scheduling method based on edge computing architecture, so as to improve the productivity of discrete manufacturing industry in the era of big data by taking advantage of edge computing.

2. Research on key technologies of edge computing in discrete manufacturing industry in the era of big data

2.1. Establishing a Big Data Multi-Cluster Edge Cloud Framework

Resource scheduling using edge computing is the migration of computationally intensive application tasks to be executed in devices with more adequate resources, thus achieving rational planning and utilization of resources and improving computational efficiency. Computational migration continuously optimizes resource utilization and improves productivity of discrete manufacturing industries by finding the optimal balance between indicators such as energy consumption, computational latency of edge devices and the amount of transmitted data [4].

To achieve the above purpose, it is first necessary to establish a multicluster edge cloud framework to solve the management and resource scheduling problems of multicluster edge clouds for discrete manufacturing industries. In this paper, we propose to build a multicluster edge cloud framework based on application latency-sensitive differences using a master-slave model. The multi-cluster edge cloud functional architecture is divided into three layers from top to bottom, central cloud, collaborative layer and edge cloud, as shown in figure 1 [5].

![Figure 1. Multi-cluster edge cloud functional architecture](image)

Multi-cluster edge clouds differentiate the latency sensitivity of applications based on domain names. When customers purchase edge cloud services from cloud provisioning vendors, they first submit edge applications and domain names to the central cloud and tag them according to the application latency sensitivity type. After receiving the application, the cloud vendor deploys the application to the edge cluster and adds the domain name used by the application to the intelligent DNS resolution server. If the application is a latency sensitive application, the domain name is resolved directly to the edge cluster. If the application is time-sensitive, domain name will be resolved to edge cluster directly. If the application is time-delay insensitive application, the domain name is resolved to the collaboration layer. After the configuration of the center cloud application to the edge is completed, the edge cloud service can be used [6].
After constructing a multi-cluster edge cloud framework, the task priority of each production link in the discrete manufacturing industry is determined on the basis of this framework in order to facilitate the efficiency of scheduling production resources in the manufacturing industry.

2.2. Discrete manufacturing task prioritization

The priority of each node in the task diagram reflects the importance of the actual production process corresponding to the node. The priority of task diagram $g_i$ can be defined according to the following formula [7].

$$\text{priority}(g_i) = \frac{d(g_i)}{CP(g_i)}$$  \hspace{1cm} (1)

In the above formula, $CP(g_i)$ is the critical path length of the current task graph; $d(g_i)$ represents the relative cutoff time of the task graph, indicating the importance of the task graph. The calculation formula for both is as follows:

$$CP(g_i) = \max_i \{EST(n_i) + t(n_i)\}$$

$$d(g_i) = \min_{n_j \in g_i} \{d(n_j) - LST(n_j)\}$$

$$EST(n_i) = \max \{EST^p(n_i), EST^{prev}(n_i)\}$$

$$LST(n_i) = \min \{LST^c(n_i), LST^{next}(n_i)\}$$  \hspace{1cm} (2)

In the above formula, $EST^p(n_i)$ denotes the condition of being in the same processor as task node $n_i$ subject to the precursor task node; $EST^{prev}(n_i)$ denotes the restriction of the children of task node $n_i$ to the current task node. $LST^c(n_i)$ represents the restriction of the child nodes of task node $n_i$ on the current task node; $LST^{next}(n_i)$ represents the limitation of the current task node that is behind the same processor as the task node $n_i$.

In each round of scheduling, in order to reasonably schedule task nodes and optimize the overall objective function, the priority of task nodes is defined in this paper. The priority $TNP(n_j)$ of task node $n_j$ can be calculated by the following formula [8]:

$$\text{TNP}(n_j) = \frac{sf}{w(n_j)}$$  \hspace{1cm} (3)

$TNP(n_j)$ responds to the flexibility of the task node, and a larger value indicates that the node is more flexible and can be placed at a later position for scheduling. $sf$ is the slack factor, which reflects the sliding space of the task node.

$$sf = LST(n_j) - EST(n_j)$$  \hspace{1cm} (4)

The task node priority $TNP(n_j)$ is sorted according to the principle of ascending order at the current moment. The task node with the highest priority (i.e., the smallest $TNP(n_j)$) is selected each time, and the appropriate processor as well as the execution slot are selected for each task node in turn. After determining the priority of the tasks in each segment of the discrete manufacturing industry, the scheduling of the generated manufacturing resources is realized.
2.3. Resource unloading and scheduling implementation

In order to avoid the low-priority tasks that exceed the time constraint from being discarded due to the high priority density, a queue preemption window function is set on the basis of the priority preemption queue to reduce the low-priority task discard rate while ensuring the real-time resource offload processing. Assuming the 1st arriving task whose priority is $a_1$, the $i^{th}$ task whose priority is $a_i$, and their arrival interval is $T_{i-1}$, and defining the parameter $T_{rest}$ to denote the first task remaining delay constraint, the task offloading should satisfy the following relation [9].

$$T_{rest} = T_i - T_{i-1};$$
$$s.t. T_{rest} \geq 0 \tag{5}$$

The relationship between task arrival time and task priority is evaluated comprehensively by introducing a priority scoring parameter $\beta_i$ within a time window, which is calculated as follows.

$$\beta_i = e^{-\delta_i^{arr-time-std}} + \ln(15 - a_i)$$
$$\delta_i^{arr-time-std} = \frac{T_i^{arr-time} - T_i^{we}}{T_i^{we} - T_i^{ws}} \tag{6}$$

In the above equation, $\delta_i^{arr-time-std}$ denotes the normalized time scale value of the current time window. $T_i^{arr-time}$ indicates the actual arrival moment of the arrival $i^{th}$ mission. $T_i^{ws}$ and $T_i^{we}$ are the actual arrival moment of the starting task and the actual arrival moment of the deadline task for the time window of the preemptive priority queue to which they belong, respectively. The time window size is set by combining task priority and arrival time, so as to develop the discrete manufacturing production task unloading order. When allocating task resources to the discrete manufacturing industry, it is necessary to meet the basic requirements such as real-time and productivity for each production segment of the manufacturing industry[10]. Therefore, the following resource scheduling objective function and scheduling constraints are established.

$$\min_{\forall n \in N} \max_{\forall m \in M} \left( \min_{\forall n \in M} \frac{D_m^n}{P_m^n \times r_m^n} + \frac{C_m^n}{Q_m^n \times q_m^n} \right)$$
$$\sum_{\forall n \in N} \sum_{\forall m \in M} I_m^n \times P_m^n \leq P$$
$$\sum_{\forall n \in N} I_m^n \times Q_m^n \leq Q_m$$
$$\sum_{\forall n \in N} I_m^n \times D_m^n \geq D_m$$
$$\sum_{\forall n \in N} I_m^n \times C_m^n \geq C_m \tag{7}$$

In the above equation, $r_m^n$ is the operational processing rate. $I_m^n$ is a directive function with business $n$ relevant edge nodes marked as 1 and non-relevant as 0. $P$ is the number of communication modules of the edge node. $Q_m$ is the computational rate of the computing unit. $Q_m$ is the number of computing resource units. $D_m$ for data processed by individual operations. $C_m$ is the number of allocated computing tasks for edge node $m$. The above resource scheduling planning problem is solved to obtain the discrete manufacturing resource scheme. Thus, the research on the method of discrete manufacturing resource scheduling using edge computing architecture is completed.
3. Simulation Experiments
This section evaluates the performance of the discrete manufacturing resource scheduling approach based on the edge computing architecture proposed above by using real data sets and comparing it with the traditional resource scheduling approach in various aspects.

3.1. Experimental Content
The method chosen for comparison is the traditional cloud-based model of manufacturing resource scheduling method. The experiments mainly compare the impact of the number of edge nodes and the stability of the method on the experimental results when both methods process the same set of tasks. The experimental data corresponding to each impact index is analyzed to draw final experimental conclusions.

3.2. Experimental results
The experimental results are shown in figure 2 below, where figure 2(a) shows the comparison of the influence of the number of edge nodes; figure 2(b) shows the comparison of the influence of the stability of the method.

Figure 2. Experimental results

Figure 2(a) gives the effect of the change in the number of task graph nodes on the NTR of the proportion of task graphs whose metrics have not been exceeded. As the number of task nodes increases, the NTR of both scheduling methods decreases, however, the performance of this paper's method is better than that of the traditional method, and the decreasing rate of this paper's method is smoother compared with the traditional method. The analysis in figure 2(b) shows that with the change of time, the proportion of untimed tasks of this method fluctuates around the mean value of 0.78, while the proportion of untimed tasks of the traditional method fluctuates around the mean value of 0.2, i.e., this method has long-time stability under the premise of ensuring the normal operation of the manufacturing system. In conclusion, this paper studies the discrete manufacturing resource scheduling method based on edge computing architecture with better application effect.

4. Conclusion
At present, a large number of discrete manufacturing systems are limited by the incompleteness of data, and the overall equipment efficiency and other index data calculation is relatively coarse and difficult to use for efficiency optimization. Edge computing, on the other hand, provides data aggregation and collection and intelligent decision analysis capabilities at the edge for discrete manufacturing. In this paper, we propose a resource scheduling method for discrete manufacturing based on edge computing architecture, which meets the requirements of discrete manufacturing for real-time, and improves the efficiency and management of each link.
References

[1] Kalsoom Tahera, Ramzan Naeem, Ahmed Shehzad, UrRehman Masood. Advances in Sensor Technologies in the Era of Smart Factory and Industry 4.0. [J]. Sensors (Basel, Switzerland), 2020, 20(23): 6783-6783.

[2] Wang M, Zhang Z, Li K, et al. Research on key technologies of fault diagnosis and early warning for high-end equipment based on intelligent manufacturing and Internet of Things [J]. The International Journal of Advanced Manufacturing Technology, 2020, 107(3):1039-1048.

[3] Heiner Reinhardt, Jan-Peter Bergmann, Marc Münnich, et al. A survey on modeling and forecasting the energy consumption in discrete manufacturing [J]. Procedia CIRP, 2020, 90: 443-448.

[4] Zhiheng Zhao, Peng Lin, Leidi Shen, et al. IoT edge computing-enabled collaborative tracking system for manufacturing resources in industrial park [J]. Advanced Engineering Informatics, 2020, 43:101044.

[5] Jakob Zietsch, Lennart Büth, Max Juraschek, et al. Identifying the potential of edge computing in factories through mixed reality [J]. Procedia CIRP, 2019, 81: 1095-1100.

[6] Anderson Carvalho, Niall O’Mahony, Lenka Krpalkova, et al. Edge Computing Applied to Industrial Machines [J]. Procedia Manufacturing, 2019, 38: 178-185.

[7] Zietsch Jakob, Vogt Marcus, Lee Benjamin D. et al. Enabling smart manufacturing through a systematic planning framework for edge computing [J]. CIRP Journal of Manufacturing Science and Technology, 2020, 31: 351-369.

[8] Jenny L. Diaz C., Carlos Ocampo-Martinez. Energy efficiency in discrete-manufacturing systems: Insights, trends, and control strategies [J]. Journal of Manufacturing Systems, 2019, 52: 131-145.

[9] Shashi Bhushan Jha, Radu F. Babiceanu, Remzi Seker. Formal modeling of cyber-physical resource scheduling in IIoT cloud environments [J]. Journal of Intelligent Manufacturing, 2019, 31(2):1149-1164.

[10] Amandeep Noliya, Sanjeev Kumar. Performance Analysis of Resource Scheduling Techniques in Homogeneous and Heterogeneous Small Cell LTE-A Networks [J]. Wireless Personal Communications, 2020, 112(4): 2393-2422.