



Edge IOT Proxy Gateway Task Scheduling Method for PIOT Service

Jichuan Zhang¹, Peng Wang¹, Shen Guo¹, Jiayin Lin¹, Sijun Qin¹, Chuanyu Tan¹, and Peng Lin²(✉)

¹ China Electric Power Research Institute, Beijing 100192, China

² Beijing VectInfo Technologies Co., Ltd., Beijing 100089, China
linpeng@vectinfo.com

Abstract. With renewable energy becoming the mainstream of the development of the world energy field in the future, virtual power plant has become a new technology in the field of renewable energy. In the traditional management system, the cloud server is responsible for handling the service needs generated by users, which makes the network face the problem of high latency and large computing pressure, which makes a large number of delay sensitive services unable to be well implemented. At the same time, the scarcity of computing resources makes edge nodes may face computing needs that exceed their own capabilities. This paper proposes an edge server collaborative computing model, which comprehensively utilizes the computing power of service terminals and edge servers, coordinates the unloading rate of computing tasks between different devices, and minimizes the delay and energy consumption. This paper proposes a load balancing model, which flexibly adjusts the task executors to maximize the efficiency of the system. Finally, we simulate the network load balancing model based on the above model. The results show that the system delay, system energy consumption and load quantity should be comprehensively considered in the unloading process.

Keywords: Virtual power plant · Mobile edge computing · Computation unloading · Delay sensitive · Load balancing

1 Introduction

With new energy becoming the mainstream of future world energy development, virtual power plant has become a local multi-energy accumulation mode to realize the large access of new energy generation to grid [1]. In the development process of this technology, IOT terminals have gradually been applied in virtual power stations [2].

As shown in the Fig. 1, the business terminal is connected to the network by the end of the device terminal arranged in the substation, and the scheduling engine in the edge attached terminal device schedules the processes in the control system to balance the containers in the system. At the same time, the scheduling engine also supports various business systems, APP, which can map the APP to different processes. The data of the edge IOT terminal can be transmitted to the business system through the IOT control platform, or directly connected with the business system to transmit data.

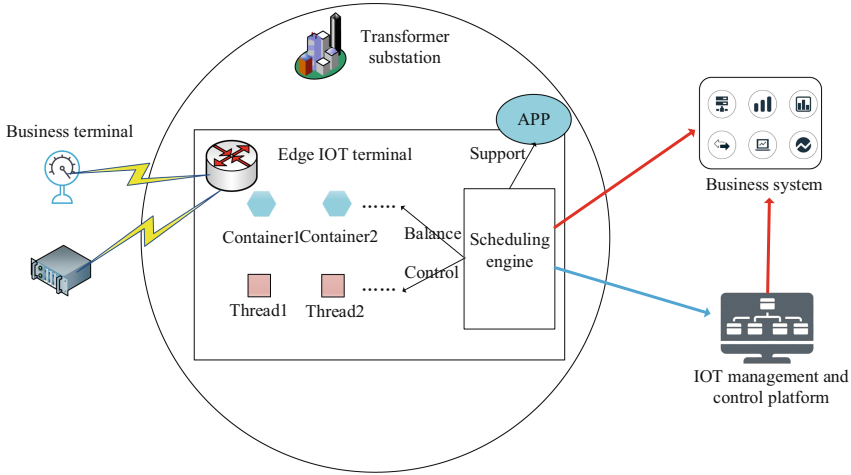


Fig. 1. Scene of edge IOT proxy gateway task scheduling for PIOT service

When data is exchanged and processed, the various resources consumed by business in cross-domain situations are quite different, so it is difficult to match the corresponding resources for business, and it is difficult to achieve a uniform processing of resource allocation [3]. In the past, management systems used to allocate business to cloud computing platforms, which put pressure on networks and servers and made time-delay sensitive tasks difficult to meet. Although edge computing can relieve the pressure of network, edge servers are prone to delay data computation due to their limited processing capacity [4].

Cloud servers and edge nodes can collaborate to accomplish tasks. This makes it possible to collaborate on computational uninstillation tasks. His important issue was to find a balance between the consumption of data processing and network transmission [5]. Therefore, this paper proposes a edge resource collaboration mechanism, which uses load factor to seek balance of task unloading in the network, so as to ensure reliable and efficient service.

We designed a collaboration model of edge nodes. It integrates the processing power of terminals and edge servers. At the same time, it reduces processing time and consumption by adjusting the unloading rate of network node. In addition, we designed a load balance model to transfer business dynamically between different nodes to improve efficiency.

In order to solve the problem of task unloading and resource allocation when there are different user terminals and servers under edge conditions, we have carried out a series of studies to minimize the energy consumption and delay of the system. This paper mainly has the following work:

1. Aiming at the problems of low computing efficiency of the user equipment terminal and excessive transmission delay of the cloud platform, we study an edge server cooperation model which will set the computing unloading rate between the end user equipment and the MEC server in the training process.

2. Due to the lack of computing ability of each edge server, this paper proposes a load balancing model, which migrates tasks between over load and relaxed edge servers to achieve the highest efficiency of the overall system. As a computing carrier, edge nodes will have limitations on computing resources and storage resources, Therefore, in the unloading process, the system efficiency is maximized without exceeding the overall limit, and the relative optimal solution of the load balancing model is found in the simulation process.
3. For different subtasks, this paper designs a task allocation mechanism between subtask domains, and makes unloading decisions for different tasks by calculating unloading, so as to improve the reliability and efficiency of the whole system.

This article is divided into five sections. Section 1 is the introduction, which briefly explains the content of the study. Section 2 is the model construction for calculating unload and resource allocation. The calculation model, collaborative calculation model and load balancing model are built successively. Section 3 is the algorithm design and implementation. In this paper, A3C algorithm is used to solve the problem that continuous and discrete motion spaces exist simultaneously in the system. Section 4 is the simulation part. In this section, two models using A3C algorithm, edge server collaboration model and load balance model, are simulated.

2 System Model

2.1 Computation Model

There will be different unloading strategies in the process of computing and unloading in the business terminal. Similar computing strategies will share the same model. Terminal obeys one task unloading strategy, that is, each task can be unloaded and executed locally, you can also perform remote uninstall on the edge server [6]. We use $a_i \in \{0, 1\}$ here to indicate that the i -th task is executed locally or on the edge server.

Local Computing

If a CPU frequency is set to f_i^{local} when a business terminal is working, the energy consumed for local data processing can be described as

$$p_i^{local} = k_1 \left(f_i^{local} \right)^3 \quad (1)$$

As you can see from the above, the local power consumed computed by the business terminal is as follow:

$$E_i^{local} = p_i^{local} \Delta t = k_1 \left(f_i^{local} \right)^3 \Delta t \quad (2)$$

So, in local computing, the data size calculated and processed by the service terminal is as follow:

$$D_i^{local} = \frac{f_i^{local} \Delta t}{c} \quad (3)$$

where c is the data processing density.

Edge Computing

After data processing tasks are generated, the terminal can transfer these tasks to edge nodes for execution. Make the computing power of edge nodes f_k^{edge} , and the energy it takes to process the data can be expressed as:

$$p_j^{edge} = k_2 \left(f_j^{edge} \right)^3 \quad (4)$$

So, the energy consumed by the edge server in handling one-off computing tasks is:

$$E_j^{edge} = p_j^{edge} \Delta t = k_2 \left(f_j^{edge} \right)^3 \Delta t \quad (5)$$

The data size processed by the edge calculation can be described as:

$$D_j^{edge} = \frac{f_j^{edge} \Delta t}{c} \quad (6)$$

2.2 Collaborative Computation Model

Compared with cloud server, the resource stock of edge servers is extremely scarce. With the increasing number of service terminals in the network, a single edge server cannot perform a large number of computing tasks [7]. The operation of edge servers in the network is different. For example, when an emergency occurs, a large number of service terminals transmit relevant data to the edge server for processing. At this time, some edge servers need to process a large number of computing tasks. At the same time, most edge servers are in a relaxed state and perform fewer tasks. Therefore, we carry out collaborative computing research to solve the problems of resource utilization and system delay, and allocate the tasks of servers with more tasks to adjacent idle servers.

When local unloading computing happens, local edge server n_i will calculate the task T_k directly. We use L_t to express the CPU cycle of execution of task T_k , and it is described as:

$$L_t = D_t X_t \quad (7)$$

Let Q_t calculate the number of CPU cycles consumed per unit of task for the edge server, which is related to the application running on the server, and this parameter can also be measured offline. f_R represents the edge server's data processing capabilities and meets $f_R \leq f_{\max}$. The data processing speed $R_l(t)$ of the local edge server is:

$$R_l(t) = \frac{f_R}{Q_t} \quad (8)$$

Computing tasks occurring in business terminals follow a Poisson distribution. They will be transferred to the edge server n_i at λ_i rate. These edge servers are modeled as M/M/1 queues. The process in which the task is transferred to edge server n_i and processed data

follows a Poisson process with arrival rate λ_i . Local Edge Server data processing time $\Delta T_l(t)$ is:

$$\Delta T_l(t) = \frac{1}{\mu_l - \lambda_i} \quad (9)$$

where $\mu_l = 1/T_{exe,l}(t)$. $T_{exe,l}(t)$ represents the last time when the local edge node completes the computing task T_k , and it can be described as:

$$T_{exe,l}(t) = \frac{D_t X_t}{f_R} = \frac{L_t}{f_R} \quad (10)$$

Combined with the above analysis of data processing, the total time consumed in data processing is:

$$T_{tot}(t) = (1 - a_n(t))\Delta T_l(t) + a_n(t)T_{tot,ij}(t) \quad (11)$$

Service delay model is shown as above. For the system energy consumption model, according to the formula, we can know the transmission rate of UAV terminal uploading to edge nodes and the transmission rate of cooperative cooperation between edge nodes. Model 1 performs unloading calculation for the local edge server. In the local unloading process, task T_k is computed by the local edge server n_i .

$$\Delta E_i(t) = \frac{1}{\mu_i - \lambda_i} \cdot P_{power} \quad (12)$$

2.3 Load Balancing Model

Edge nodes assume different tasks, and their computing resources are not evenly configured in the network. Therefore, when choosing the execution of the computing task, the resource configuration of each edge server needs to be considered to ensure that the computing resources of the edge server can be fully utilized and network nodes can use this to achieve load balancing. This model evaluates the current state of an edge server by managing its storage and computing resources jointly. Make the data processing capacity of edge server n_i as $W_{cap,i}(t) = \{E_{cap,i}(t), C_{cap,i}(t)\}$, where $E_{cap,i}(t)$ and $C_{cap,i}(t)$ are the storage and computing resources of edge server n_i . Edge server n_i is in $F_{cur,i}(t) = \{E_{cur,i}(t), C_{cur,i}(t)\}$ state, $E_{cur,i}(t)$ and $C_{cur,i}(t)$ represent the storage and computing resource usage of edge server n_i . Edge Server n_i 's storage resource usage $E_i(t)$ and computing resource usage $C_i(t)$ can be represented as:

$$E_i(t) = \frac{E_{cur,i}(t)}{E_{cap,i}(t)} \quad (13)$$

$$C_i(t) = \frac{C_{cur,i}(t)}{C_{cap,i}(t)} \quad (14)$$

Further, the running state of edge server n_i is:

$$F_{load,i}(t) = \beta E_i(t) + \eta C_i(t) \quad (15)$$

where $\beta + \eta = 1$, β and η represent the weight of their indicators, and the higher the weight value, the more specific resources it consumes and the stronger its dependency.

Set $F_s = \{F_{load,1}(t), F_{load,2}(t), \dots, F_{load,n}(t)\}$ the load rate for all edge servers in the network. From this parameter, you can see the running state of the whole network. When load rate becomes rather lower, the better the overall performance of the network without too much pressure. The average server load rate $F_{ave}(t)$ is described as:

$$F_{ave}(t) = \frac{1}{n} \sum_{i=1}^n F_{load,i}(t) \quad (16)$$

Set σ_F the standard deviation of the server load rate. This parameter indicates the overall stability of the network. The lower the standard deviation, the more stable the whole system is. It is described as:

$$\sigma_F = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (F_{load,i}(t) - F_{ave}(t))^2} \quad (17)$$

So, the load factor δ_i is used to represent the running state on the edge server, it can be described as:

$$\delta_i = \frac{F_{load,i}(t) - F_{ave}(t) + \sigma_F}{2\sigma_F} \quad (18)$$

3 Design and Implementation of Algorithm

The neural network optimizer inside the A3C algorithm uses an asynchronous gradient descent algorithm. There are both continuous and discrete states in our model, so the A3C algorithm is an excellent problem solving algorithm.

3.1 State Space and State Transition Probability

This model integrates computing ability $C_s(t) = \{C_i(t)|i = 1, 2, \dots, n\}$ and the storage ability $E_s(t) = \{E_i(t)|i = 1, 2, \dots, n\}$, and communication resources $G_2(t) = \{g_{i,j}(t)\}$ of edge servers to describe the state space:

$$S_{s2}(t) \triangleq \{C_s(t), E_s(t), G_2(t)\} \quad (19)$$

When one action is executed, the probability of moving from state to another state can be described as:

$$\Pr(s_{s2}(t+1)|s_{s2}(t), a_{s2}(t)) = \int_{s_{s2}^t}^{s_{s2}^{t+1}} f(s_{s2}(t), a_{s2}(t), s_{s2}) ds_{s2} \quad (20)$$

3.2 Action Space

Action space includes unload decision $a(t)$ and unload rate decision $\alpha(t)$. We use $A_{s2}(t)$ to define the set of actions: $A_{s2}(t) \triangleq \{a(t), \alpha(t)\}$. Where $\alpha(t) \in [0, 1]$ [0, 1], and $a(t)$ is described as:

$$a(t) \triangleq \{a_1(t), a_2(t), \dots, a_N(t)\} \quad (21)$$

3.3 Reward Function

Our algorithm uses the following reward function to reduce system latency:

$$r_{s2}(t) = \begin{cases} W_{s2}(t) \\ 0, & \text{otherwise} \end{cases} \quad (22)$$

where $W_{s2}(t) = \frac{1}{\sum_{n=1}^N T_{tot,n}}$.

4 Simulations

We propose two different models with A3C algorithm in this study: the model of collaborative computing and the model of load balancing. The first is the uninstalation mode between business terminals and edge nodes, that is, the formulation of the unloading strategy between the business terminal and the edge server. If a large number of computing tasks are performed by the business terminal, it will lead to excessive pressure on the business terminal and unreliable system. The second is the edge server collaboration unload model, it includes the computational unload collaboration between the edge servers. This load rate balances the overall system delay and energy consumption, minimizing overall system consumption. Both of the above unloading processes involve the determination of the unloading rate.

Figure 2 shows an unload delay simulation in which the horizontal coordinate is the unload rate, one Y axis represents the average reward, and the other Y axis represents the average unload time cost. It can be seen from the figure that when the unloading rate increases, the time consumption first decreases and then increases, while the average number of rewards first increases and then decreases. The following figure shows that when the uninstalation rate reaches 0.65, we can get the least time delay, The system becomes stable and efficient.

Figure 3 shows the energy consumed by the system, including two Y axes, one represents the average time consumed by the system and the other represents the energy consumed by the system with the change of unloading rate, this picture shows the change of system energy consumption and average return, in which the former decreases first and then increases, and the latter increases first and then decreases. From Fig. 3 we can see when the uninstalation rate reaches 0.85, system will consume the least energy.

By synthesizing the above unload rates and edge node load models, it is found that the optimal unload rates differ for different system models. Therefore, in the process of

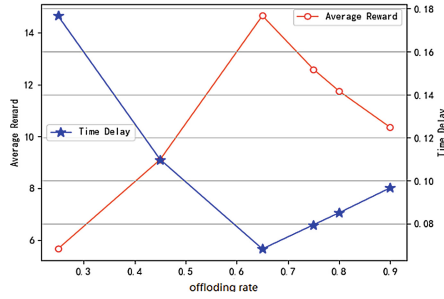


Fig. 2. System delay

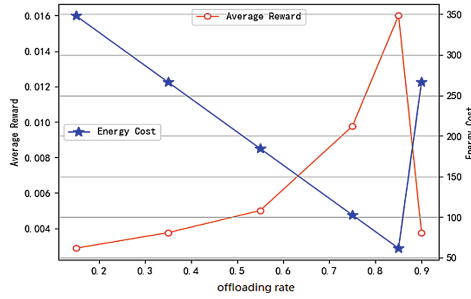


Fig. 3. System energy consumption

system operation, we need to comprehensively consider two aspects to ensure that the time and energy consumed by the system are as small as possible. In our proposed model, when the number of edge servers and business terminals is the same, the system consumes the least time and energy. However, in order to improve the system performance, we can try to increase the number of loads in the system within a certain range to make the system more efficient. In summary, the above three factors should be considered in the process of unloading to calculate the unloading.

5 Conclusion

This paper presents a reliable collaboration mechanism for edge resources for distributed business in virtual power stations. By establishing a load balancing model and a collaborative execution model between edge servers in network, the system offload decision is made. To minimize system latency and energy consumed, we design a load balancing model, it migrates tasks between different edge servers with different running state to improve system efficiency. Finally, the algorithm we designed divides the optimization problem into two parts, local computing or edge node collaboration, through this method, we can reduce the complexity of the algorithm and can obtain a stable and convergent optimal solution in a short time. In the complex communication scenario, through the collaboration between edge servers, maximize the benefits of the system.

Acknowledgement. This work is supported by science & research project of SGCC. (Development and Application of Low Delay, Safe and Reliable Intelligent IOT Agent Device. Project No. 5700-202041163A-0-0-00).

References

1. Wei, Z., Yu, S., Sun, G., et al.: Concept and development of virtual power plant. *Dianli Xitong Zidonghua/Autom. Electr. Power Syst.* **37**(13), 1–9 (2013)
2. Yan, Z., Zeng, L., Xi, C., et al.: Study of architecture of power Internet of Things. In: IET International Conference on Communication Technology and Application, pp. 718–722 (2011)
3. Gavrilovska, A.: Resource management in wireless heterogeneous networks (WHNs). In: International Conference on Telecommunication in Modern Satellite, pp. 97–106. IEEE (2009)
4. Chen, Y., Cai, Z., et al.: An optimization method of edge computing terminal deployment and service allocation in power internet of things. *South. Power Syst. Technol.* **16**(3), 46–49 (2022). <http://dx.doi.org/10.13648/j.cnki.issn1674-0629.2022.03.013>
5. Hairuman, A., Zahra, A., Kusuma, G.P., et al.: MEC deployment with distributed cloud in 4G network for 5G success. In: International Conference on Information Technology, Computer and Electrical Engineering (ICITACEE). IEEE (2019)
6. Zhang, Y., Liang, Y., et al.: Survey on the methods of computation offloading in mobile edge computing. *Chin. J. Comput.* **44**(12), 2406–2430 (2021)
7. Stephanie, V., Chamikara, M., Khalil, I., et al.: Privacy-preserving location data stream clustering on mobile edge computing and cloud. *Inf. Syst.* **2**, 101728 (2021)