Event-awareness Algorithm in Serverless Computing

Junyao Hu

Computer Science and Technology, China University of Mining and Technology
Xuzhou, China

***Abstract*—This essay mainly introduces an event-awareness algorithm in serverless computing. This algorithm can assign the event to different processing devices such as mobile terminal, edge device and cloud server according to the type and attribute of events. We try to make cloud, terminal and edge into a powerful computing network suitable for changeable environment. This architecture can make offloading decision problems according to the environment, which is conducive to saving computing resources, improving resource utilization, and building faster and more economical network transmission services, while ensuring the security and stability of information transport. On the base of Energy Constrain Computing Offloading with Sleep Control for Could-edge-terminal Collaborative Network Algorithm (COSC), By considering more demand factors, we improved this algorithm into the algorithm to solve offloading decision problems. Then we use deep learning method to verify the efficiency of COSC in typical application environment, and compares it with other methods. Finally, we look forward to the future development of service-less computing.**

***Index Terms*—serverless computing, event-awareness algorithm, offloading decision problem, cloud-edge-terminal collaborative network, COSC, deep learning**

I. Introduction

With the rapid development of communication technology, Internet is gradually evolving from the traditional information publication platform to a distributed computing environment. Cloud computing technology opens up more and more data resources, computing resources and applications as services on Internet. [1][20]

Serverless computing, as an emerging computing model, has dramatically changed the cost model of running software applications by eliminating the overhead involved in maintaining server resources. However, lacking standardization and ecosystem maturity, the process of research and application can be more challenging due to the more dynamic change.

2019 is the first year of 5G, and edge computing is coming to the fore. With the arrival of 5G, the amount of data in edge applications increases exponentially. These data are formed and accumulated in terminals, transmitted to the cloud for data processing, and then returned to terminal guidance services. This series of actions will produce hundreds of Gb/s high demand for network bandwidth, not only there will be delays, but also need to face many problems such as weak network card, low connection success rate, user experience cannot be guaranteed. At the same time, large bandwidth will cause huge transmission pressure on the back network and service center,

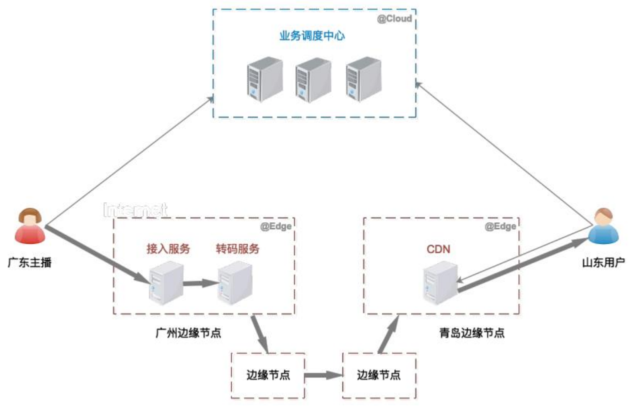

Fig. 1 Edge cloud computing interactive live broadcasting service architecture

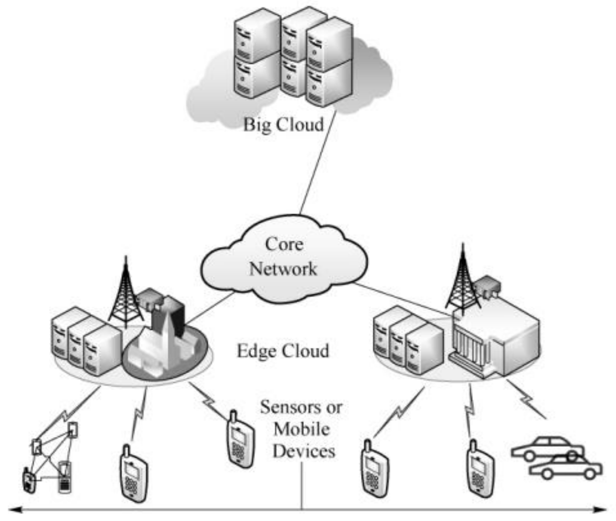

Fig. 2 Mobile edge computing (MEC) architecture

and enterprises will also face huge bandwidth costs. This means that centralized data storage and processing mode will face difficult bottlenecks and pressures. Ultimately, cost is one of the key factors that allows cloud marginalization to develop and centralized computing to become marginal again.[15][19]

Take common interactive live broadcast scenarios for example, a large number of interactions, such as real-time comments, giving tips and gifts, flash sale, etc., increase the amount of data. Meanwhile, interactive live broadcast scenarios require very accurate and fast computing capabilities. If the video computing is performed on the terminal, a large amount of computing power will be consumed, which requires high performance and power consumption of the terminal. However, if the computing is placed in the cloud center, high video transmission costs will be incurred. At this time, the terminal computing power moves up and the cloud computing power sinks, forming the fusion of computing power at the edge. The cloud-edge-terminal collaborative network architecture will play an important role.

We can think of “cloud” as the central node of traditional cloud computing and the control end of edge computing. Edge refers to the edge side of cloud computing, which is divided into infrastructure edge and device edge. “mobile” refers to terminal equipment, such as mobile phones, smart home appliances, various sensors, cameras and so on. Cloud computing can grasp the overall situation, process a large amount of data and conduct in-depth analysis, playing an important role in business decisions and other non-real-time data processing scenarios; Edge computing focuses on local areas and can play a better role in small-scale and real-time intelligent analysis, such as meeting the real-time needs of local enterprises. Therefore, in intelligent applications, cloud computing is more suitable for centralized processing of large-scale data, while edge computing can be used for small-scale intelligent analysis and local services. [1][2][3]

Choosing to calculate location and is one of the current technical difficulties, the need to turn the edge and center cloud computing and IoT connection and calculated the synergy, play edge localization computing and cloud center scale, and low cost, IoT terminal perception and so on various aspects of advantages.

II. Related Work

In this section, we introduce some related works about serverless computing, cloud-edge-terminal network researched by some cloud computing companies. Then recent advances about adaptive perceptual decision algorithm are presented.

A. Serverless Computing

The development of cloud computing has gone through several stages: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). The trend of Serverless is becoming more and more obvious. Today, Serverless Computing is becoming more and more perfect as the application of cloud native computing model. Related products are also the battlefield of various cloud manufacturers.

Serverless computing refers to the concept of building and running applications that do not require server management. It describes a finer-grained deployment model where applications, bundled as one or more functions, are uploaded to a platform and then executed, scaled, and billed in response to the exact demand needed at the moment.

Serverless computing doesn’t mean we don’t use servers to host and run code, nor does it mean we don’t need operations engineers. It means that consumers of serverless computing no longer have to spend time and resources on server configuration, maintenance, updating, scaling, and capacity planning. All of these tasks and functions are handled by the serverless platform and are completely abstracted from the developers. Therefore, developers focus on writing the business logic of the application. Operations engineers are able to shift their focus to more business-critical tasks.

B. Edge Computing

The Internet of Things is booming in all fields, and the era of the Internet of everything is drawing near. However, with the development of business and the rapid increase of IoT devices, it is gradually found that cloud-based computing cannot meet the actual needs of many scenarios.[6]

1) **Massive data have huge pressure on network bandwidth**
   The cloud center has powerful processing performance and can process massive data. However, with the development of the Internet of Things, now almost all electronic devices can be connected to the Internet, these electronic devices will generate a huge amount of data, sending these massive data to the cloud center has become a problem, which is a huge challenge to the network bandwidth.
2) **Networked devices have increased demands for low latency and collaborative work**
   The system performance bottleneck of cloud computing model lies in the limitation of network bandwidth. It takes a certain amount of time to transmit massive data and a certain amount of time for cloud center to process data, which will increase the request response time and result in poor user experience. However, in emerging IoT application scenarios, such as real-time voice translation and driverless cars, the response time requirements are extremely high, and relying on cloud computing is not practical.
3) **Connected devices involve personal privacy and security**
   A large amount of data in terminal devices will involve personal privacy, and data security risks will be greatly increased when transferred to the cloud center.

If edge computing is adopted, massive data can be processed nearby, and a large number of devices can also work efficiently and cooperatively, and many problems can be solved quickly. Therefore, edge computing can theoretically meet the critical needs of many industries in terms of agility, real-time performance, data optimization, application intelligence, and security and privacy protection.

The difference between cloud computing and edge computing mainly lies in that cloud computing has the most computer resources and is suitable for applications requiring high computing resources, such as training a neural network for image recognition; Edge computing has slightly fewer computing resources, but because of its prime location, it is suitable for high-latency applications, such as using the model you just trained to identify the person in front of you.

C. Cloud-edge-terminal Collaborative Network

Cloud computing, edge computing and terminal services all have their own advantages. Rational allocation of resources and rational invocation of computing services are the current research direction. The following are some of the research results that have been carried out by Chinese companies.[6]

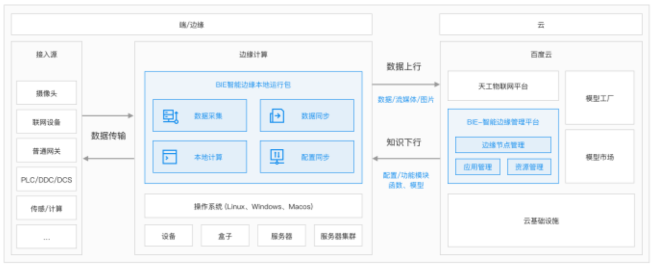

Fig. 3 Baidu BIE computing platform

1) **Alibaba**
It launches Link IoT Edge platform. It can be deployed on smart devices and end-to-end compute nodes of different scales. By connecting devices with different protocols and data formats through the object model, it provides secure, reliable, low latency, low cost and easy to expand local computing services.

2) **Tencent**
It launched CDN Edge, which sinks the service of data center to the Edge node of CDN, corresponding end users with the lowest delay, and at the same time reduces the computing pressure and network load of user data center.

3) **Baidu**
It launched the intelligent Edge BIE to extend the cloud computing capability to the user site and provide temporary off-line and low-delay computing services. At the same time, with the intelligent edge cloud management suite, an end-to-end cloud integrated solution of "cloud management and end-to-end computing" was formed.

4) **Huawei**
It launched the IEF platform in 2018. By managing users' edge nodes, Huawei provides the ability to extend cloud applications to the edge, link edge and cloud data, and provide enterprises with a complete edge computing solution with integrated services of edge and cloud collaboration.

5) **China Telecom**
China Telecom operators rely on 5G to fully deploy MEC. Mobile edge computing (MEC) uses wireless access network to provide services and cloud computing functions needed by telecom users and realize flexible utilization of computing and storage resources. Multi-access edge computing (MAEC) extends edge computing from telecom cellular networks to other wireless access networks.

D. Demand Indicators

In order to comprehensively analyze the applicability of the three calculation methods, we select different demand types as the standard of data quantification.[2][4][18][19]

1) **Computing power**
The development of 5G network has increased the volume of data, posing a greater challenge to the computing power of cloud computing. For example, when there is a bottleneck in terminal computing power, the structure of the traditional operating system makes the user experience completely influenced by the quality of the terminal. Reducing the pressure on data processing requires a new framework for cloud computing, the value of which is to provide computing like electricity, reducing the pressure on data center computing power and storage, and network bandwidth transmission. Traditional cloud computing uploads all data to the cloud computing center over the network, but this centralized data center processing mode puts pressure on bandwidth and data center computing power and bandwidth.

2) **Delay requirement**
The time spent from generating tasks to obtaining the results of processing tasks, including transmission delay, calculation delay, queuing delay, etc. Edge computing devices in the IoT can process data locally or in a nearby edge data center, which can greatly improve the response speed of smart devices. For example, in the field of face recognition, compared with cloud computing, the response time of edge computing is reduced from 900ms to 169ms, even faster than the response time of human face recognition (370-620ms).

3) **Data capacity**
Liang Hua, chairman of Huawei, said at the China Mobile Global Partnership Conference in 2021 that the total amount of global information data is showing explosive growth. In 1992, all mankind produced only 100GB of data every day. Today, the world's 7 billion people generate 1.5 GB of data per person per day on average. A single self-driving car can generate 64 TB of data per day. As more and more devices connect to the Internet and generate data, cloud computing with a central server as its node can run into bandwidth bottlenecks.

4) **Cost**
Many companies have not done a better job of saving costs because of inexperience or poor planning. ParkMyCloud, a cloud resource scheduling service, estimates that companies will waste $21 billion a year on cloud computing between now and 2021, or nearly $2.4 million an hour. Firstly, the billing strategy should be formulated according to the cost of various services, supply and demand relationship and other factors. Second, collect billing receipts, such as the hardware resources used, network services, etc., to calculate service fees. In order to realize the corresponding change of policy automatically according to the change of real-time service, we can further enrich the flexibility of charging policy, and use the rate strategy of ladder charging. The unit of measurement can be time period or resource usage.

5) **Energy consumption demand**
The energy consumed by the equipment when performing a task, including calculation energy consumption, transmission energy consumption, etc. High power consumption caused by high data center load is also a core problem in data center management planning. Power consumption is important for each layer of edge computing architecture, and most end-device performance is limited by the design of batteries. In general, the power consumption of a device is approximately linear depending on its CPU load. Considering the difference between power consumption and power consumption, the

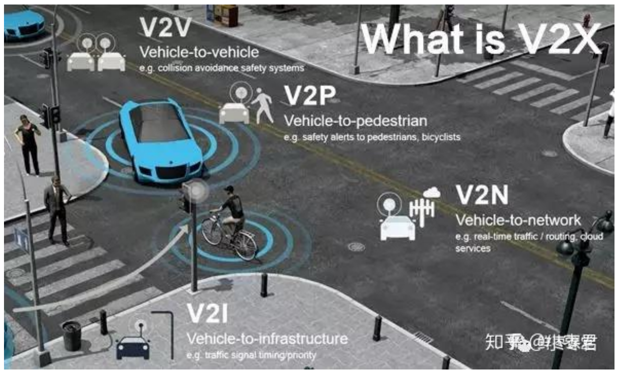
What is V2X
V2V
Vehicle-to-vehicle
e.g. collision avoidance safety systems
V2P
Vehicle-to-pedestrian
e.g. safety alerts to pedestrians, bicyclists
V2N
Vehicle-to-network
e.g. real-time traffic / routing, cloud services
V2I
Vehicle-to-infrastructure
e.g. traffic signal timing/priority

Fig. 4 Internet of vehicles requires vehicles to quickly sense the environment to ensure human safety

power consumption also depends on the amount of time the task takes to execute. In terms of the overall power consumption of the system, it is beneficial to offload the task to a device with a faster computing speed.

6) **Quality of service (QoS)**
In the IoT environment, the mobility, heterogeneity, instability and other characteristics of devices pose challenges to providing reliable QoS. In the unloading process of computing, limited communication and computing resources need to be reasonably allocated to maximize the use of network resources to ensure that user experience will not be greatly affected. The measurement criteria of QoS include service reliability, fairness and quick response capability in heterogeneous environment. Li et al. proposed a statistical calculation model and a statistical transmission model to quantify the correlation between QoS and task unloading strategy.

7) **Security**
Edge computing will disperse data processing, storage and application in a wide range of devices and data centers, so it is difficult for a single attack to destroy the entire network; On the other hand, traditional cloud computing takes a long path to transmit the private data collected by wearable, medical and industrial devices to the data center, which is prone to data loss or information leakage. By saving and processing data at the edge, this risk can be effectively avoided. In addition, ownership of the data collected will be transferred from the service provider to the end user.

E. Application Scenarios

Different application scenarios have different quality of service requirements. Therefore, we select the following typical application scenarios to analyze demand indicators which are essential to current using environment. [8][9][11]

1) **Intelligent transportation**
It should support unmanned driving, traffic control and other applications in the future. This scenario should strictly meet the constraints of time delay to ensure safety. At the same time, a balance needs to be struck between reducing energy consumption and air pollution while ensuring a comfortable driving experience.

2) **Intelligent medical treatment**

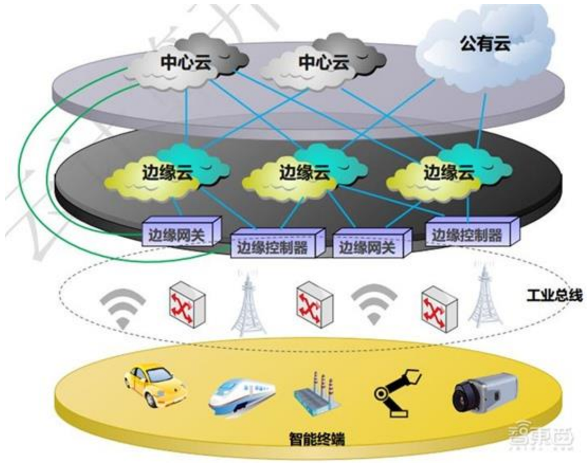

Fig. 5 Smart Industrial on cloud-edge-terminal network

This system provides intelligent access to medical equipment to provide effective treatment for patients. The patient's personal device connects to the medical terminal in real time, and the doctor diagnoses the patient remotely. In this case, the main demand is time delay, especially in the observation of emergency diseases, medical information uploading and medical diagnosis require ultra-low time delay information transmission and processing. At the same time, the energy limitation of sensors and other devices should be considered in this scenario to prevent the overuse of daily node energy and the delay of emergency rescue due to insufficient energy.

3) **Intelligent disaster relief**
In disaster scenarios, infrastructure damaged, the damage to electric power, people need help, who carry the smart phones, electricity equipment such as torches, radios and computers became the self-help equipment, considering the equipment energy limited is more important, to complete response equipment under the energy limit rapid response.

4) **Intelligent grid**
With the goal of saving power resources, smart grid manages energy supply for urban residents, and time delay and energy consumption demand can be flexibly adjusted. For example, the smart distribution service in the power grid includes intelligent fault monitoring and other links. The precise control of distribution network management and control, micro-synchronous measurement and other services requires an ultra-low delay of less than 10ms. In addition, residential energy pricing strategies used by energy operators limit the supply of energy to equipment.

5) **Smart home**
Smart home appliances are mainly composed of intelligent robots, intelligent lighting, intelligent switches, security monitoring, etc., mainly achieved by sensors, mobile phones, personal computers, wearable devices, VR, AR devices, electronic appliances and a large number of resource-constrained devices. For example, in a mobile RFID system, the energy of the terminal device is limited, and the RFID reader must supply power for all the passive tags in its query area. If the RFID reader processes too many tasks and consumes too much energy, it may run out of energy and lose information. These resource-limited

devices need to upload real-time messages and support intelligent applications with limited energy supplies.

6) **Intelligent education**
Smart education systems tend to provide a set of malleable educational devices that not only focus on the current situation, but also consider the future situation. Due to the increasing number of students, the stability and QoS are highly required.

III. . System Description and Modeling

It is an important and challenging task to allocate the computationally intensive tasks to the edge of the network reasonably, and use enough computing resources to carry out the computation processing, and then return the computation result to the terminal device. By studying other researcher's work, we try to create a new event-aware algorithm called Energy Constrain Computing Offloading with Sleep Control for Could-edge-terminal Collaborative Network Algorithm (COSC), it is based on COSC proposed by Q. Liu.[4]

A. COSC Algorithm Introduction

COSC firstly establishes the delay energy consumption model of each network segment at the cloud side based on related services, and uses the combination of load rate and hibernation threshold to control server hibernation to save energy consumption, proposes the optimization problem of long-term average energy consumption of the system, and then designs the computational unloading problem model based on server hibernation. Then, the model was solved based on Lyapunov stochastic system optimization theory. Considering the time dimension information of tasks, a cross-time energy management model was designed based on energy consumption queue, and the long-term energy consumption optimization problem was transformed into a time-slot unloading optimization problem. Moreover, the queuing delay is calculated by the energy consumption queue, which extends the standard Lyapunov technology and allows the task length to be longer than one time slot to meet the diversified computing needs. Finally, considering that the traditional convex optimization method usually requires high computational complexity to obtain the near-optimal solution, and it is difficult to involve the information of task time dimension, the reinforcement learning method is considered to solve the computational unloading decision problem, and the near-optimal solution is made according to the experience obtained from the environmental information.

B. System Modeling

The system is divided into time slots in the time domain. The time slot set is represented by $t \in \{\, 1, 2, \dots, T\,\}$, and each time slot has length l. The terminal layer composed of various video surveillance, smart home, industrial building and electric vehicle terminal equipment generates various task data to be processed in each timeslot. The data is generated in the $t-1$ timeslot, and then the unloading decision is made from the resource allocation module of the cloud in the t timeslot. Unloading task generated by the event to the local terminal, edge node, and cloud platform for processing. It should be noted that the execution time required by different tasks is different, that is, some tasks may not be completed within the time length l of one timeslot and will continue to be processed in the next timeslot. Therefore, queuing delay must be considered when calculating delayed tasks if the tasks in the previous time slot have not been completed.

$U_{m,i}$ stands for terminal m while links to edge nodes i, and M_i is the number of terminals which U_i has. If task $R_{m,i}$ is made by terminal $U_{m,i}$ when timeslot t by using Poisson process, R_i stands for new tasks requests which made by U_i . $R_{m,i}\left(L_{m,i}^t, X_{m,i}^t, \tau_{m,i}^t\right)$ stands for a calculation task. $L_{m,i}^t$ is the volume of data (bit), $X_{m,i}^t$ is event processing density and $\tau_{m,i}^t$ is execution time limit, the calculation task $R_{m,i}$ should finish successfully in limited time $\tau_{m,i}^t$. Terminals can generate different types of tasks. These tasks vary in data size and computing requirements, measured in CPU cycles. In order to simplify the system model, we assume that the unit input task is $R(L, X, \tau)$, while other expected input tasks are multiples of R, also $R_{m,i} = C_{m,i}R$. Within each timeslot, the resource allocation module provides unload policies that determine whether tasks are handled by terminals, edge nodes, or clouds. We use $x^h = \{0,1\}, (h = devide, edge, cloud)$ to indicates an unloading policy. Therefore, the unloading policy must meet the following operation constraints:

$$x^{device} + x^{edge} + x^{cloud} = 1 \qquad (1)$$

OF stands for an unloading vector, and the vector of $R_{m,i}$ is $OF_{m,i} = \{x_{m,i}^{device}, x_{m,i}^{edge}, x_{m,i}^{cloud}\}$.

Through literature reading and summary of general life and study experience, we conclude the following empirical formula for the next step of calculation about terminal model, edge model and cloud model.

C. Terminal Model

1) **Delay requirement**
The delay in processing computing tasks is related to x and can be expressed as:

$$D_l^{m.i} = W_{m,i}^l / f_{m,i}. \qquad (2)$$

$O_{m,i}(t)$ stands for current remain energy. The queuing delay of tasks is:

$$D_q^{m.i} = O_{m,i}(t) / k f_{m,i}^3. \qquad (3)$$

2) **Energy consumption demand**
In every timeslot, if $x^{device} = 1$, the event will be processed by mobile terminal itself, without upload any information to edge network or could network. Depending on different application, finishing a task $R_{m,i}$ needs CPU cycles $W_{m,i}^t = L_{m,i}^{device} X$. In this passage, we use $f_{m,i}$ to describe the CPU working cycle time of terminal $U_{m,i}$, and the maximum of CPU cycle is $f_{\max}$. In this model, the main energy consumption is made by CPU operations. So, in a timeslot the energy consumption can by expressed as:

$$p_l^{m,i} = k L_{m,i}^t X f_{m,i}^2. \qquad (4)$$

The number k is the effective switching capacitance related to the chip structure. The union of terminal U_i has a total energy consumption:

$$p_l^i = \sum_{m=1}^{M_i} p_l^{m,i} = \sum_{m=1}^{M_i} kL_{m,i}^t X f_{m,i}^2. \tag{5}$$

D. Edge Model

1) **Delay requirement**

Terminal-edge wireless transmission delay is generated during the unloading process. Terminals send computing tasks to edge nodes through the uplink channel, and task data may be uploaded at any timeslot after the mobile device sends task-related information to the base station. P_i^{TX} is the uplink transmission power, then the uplink transmission rate between terminals and edge nodes is obtained by Shannon theorem. Therefore, the transmission delay cost of edge computing is:

$$D_{tx}^{m.i} = L_{m,i}^l / r_{m,i}. \tag{6}$$

So the total transport cost is:

E. Cloud Model

1) **Delay requirement**

In every timeslot, if $x^{cloud} = 1$, the event task will be sent by mobile terminal to cloud. The delay time from edge to cloud is:

$$D_{bh}^{m,j} = C_{m,j} \frac{\tau}{1 - C_{m,j}\tau\psi^t}, \psi^t < \frac{1}{C_{m,i}\tau} \tag{7}$$

$\psi^t = \sum_{m=1}^{M_i} x_{m,j}^{cloud}$ stands for the number of tasks have been offloaded on the platform.

F. COSC Algorithm

COSC does not need to traverse the entire solution space when solving unloading decision and dormancy threshold problems, which is faster than traditional algorithms. To find the optimal unloading strategy and dormancy threshold using reinforcement learning, it is necessary to find the state, action and reward in the problem. The goal of COSC algorithm is to design a strategy function and gradually learn this strategy to quickly generate the optimal unload action x and hibernation threshold PI. In this paper's problem, the state is $S = \{O(t)\}$, $O(t)$ stands for which the virtual energy consumption queue, remains constant within each time slot, but can vary from one time slot to another. Actions in the network determine the offload policy x of each task and the hibernation threshold PI of the server. After receiving requests from different tasks, the controller schedules edge nodes and cloud server resources to process tasks for users. Therefore, the action space of the system can be expressed as $A = \{OF, PI\}$. OF, PI is offloading vector and hibernation threshold respectively. R stands for reward, defined by the change in KA value after a state action on (s, a) occurs. Agents in reinforcement learning are designed to maximize cumulative discount rewards over time. The reward function of the state action pair is defined as:

$$R = -\left(KA^{after} - KA^{before}\right). \tag{8}$$

With the accumulation of the number of iterations, the system can converge to the optimal state, in which the value KA remains unchanged and remains at the minimum.

COSC uses the agent to interact with the environment, which is composed of the cloud edge and end integrated network facilities to provide the agent with the state information required by the network. Rewards can be calculated based on state changes and fed back to the agent. Then, the resource allocation module sends the unload instruction according to the change of the status, consisting of two alternating stages: the generation and update of the unload policy and the sleep threshold. Unloading behavior and dormancy threshold are dependent on the use of neural network. In iteration, neural network mainly changes the weight parameter θ of connected hidden neurons. In the t time slot, the neural network takes the network state S={O(t)} as the input and outputs the current action based on its current policy:

$$A_t = \pi(S_t, \theta) \tag{9}$$

The network selects actions from A_t in policy $\chi-\text{greedy}$. Under $\chi-\text{greedy}$ strategy, given the state, the agent chooses the action with the highest output value of the neural network with probability $1-\chi$, or chooses an action at random with probability χ, and χ is usually a very small value. The reward r is calculated according to the figure above, and the newly acquired state action pair (s_t, a_t, r, s_{t+1}) is added to the empirical replay library, where the action-state pairs are played to train the neural network.

Finally, the algorithm in this paper reaches a stable state, that is, the reward function R cannot produce any new positive change, so as to obtain the optimal unloading strategy. After obtaining the unloading strategy, the energy queue is updated. The following algorithm summarizes the process of COSC algorithm.[4][7][13]

COSC Algorithm

```
Input: weight parameter $v$, energy consumption queue
       O($0$), neutral network parameter $\theta$, reward $R$
Output: offloading decision $OF$, sleep threshold value $PI$
Init: $O(0) \leftarrow 0$, $\theta \leftarrow random$, experience replay←null
for timeslot do
  for iter do
    look the state $s_t$, take action according $A_t = \pi(S_t, \theta)$
    take action $a_t$, calculate current reward $R(s, a)$
    update $s_{t+1} \leftarrow s_t$
    add $(s_t, a_t, r, s_{t+1})$ into experience replay
    if check() then
      random sample new dataset
      train neutral network and update parameter $\theta$
    else take action $a_t \in A_c$
  end for
  update energy consumption queue $O_{t,m}$
end for
return $OF, PI$
```

IV. Result

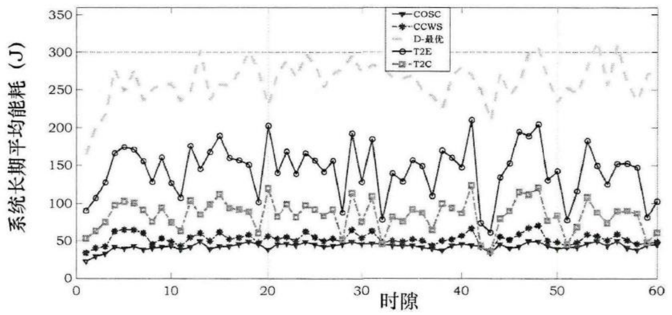

Fig. 6 Long-term energy consumption performance of the system

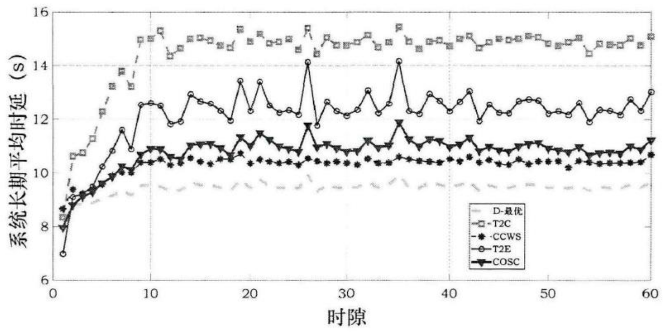

Fig. 7 Average system latency performance of the system

In order to compare the effect of this algorithm, COSC is also compared with the following four task unloading algorithms:

1) **Terminal to cloud uninstallation algorithm (T2C)**
   The edge side does not participate in the uninstallation function, and only performs computing uninstallation between the cloud data center and terminals to process tasks.
3) **Delay optimization (D-optimization)**
   Only consider the calculation of unloading delay cost to achieve the optimal, without considering the impact of energy consumption.
4) **Non-hibernation cloud side computing uninstallation method (CCWS)**
   The cloud side computing uninstallation is considered. During the process of computing uninstallation, the hibernation of edge servers is not considered.
5) **Terminal to edge unload algorithm (T2E)**
   Only unload between edge and terminal, without cloud participation.

COSC algorithm proposed in this paper, from two aspects of energy consumption management and timing optimization on the edge of cloud computing tasks for unloading, energy management in the network's overall energy consumption is reduced, and the limitation of terminal power supply, timing optimization is mainly manifested in lower overall average delay for a long time, to ensure a single task completion time within the allowed time delay. In addition, this paper observes that idle servers in the network cause energy loss, so sleep of edge nodes is added to further save energy consumption.

Shows system delay cost and energy consumption queue values from the 100th to the 150th time period. It describes how

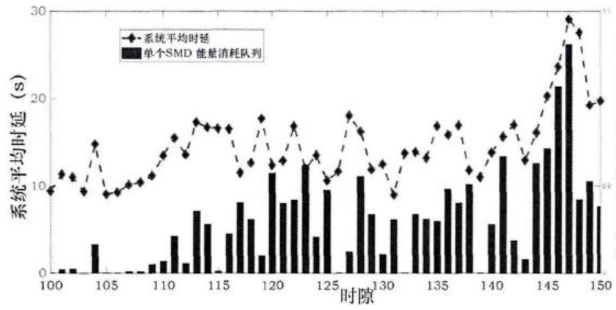

Fig. 8 Relationship between energy consumption queue and system average delay

energy consumption queues affect total latency and guide terminal unload decisions. For example, from the 110th to the 113th time cycle, the task execution time increases as the energy consumption queue increases. This is due to queueing delay caused by the task being unloaded at the user terminal. Therefore, in the next time cycle, COSC uses the energy consumption queue to influence the unloading decision, so that the unloading decision tends to reduce the task unloading to the user side, thus shortening the delay. It can be seen from the figure that the energy consumption queue will affect the cross-time slot unloading decision, so it can be regarded as the extension of the standard Lyapunov method.

V. Conclusion

In order to solve the problems of high end-to-end delay, excessive network channel pressure and high average energy consumption in traditional networks, and to meet the communication requirements of energy conservation and environmental protection, high transmission rate and low delay in smart city 5G network, COSC is designed in CETCN architecture based on dormancy control for cloud side-end collaborative network computing offloading algorithm.[11][16]

COSC cloud edges is used to calculate unloading complete CETCN framework across the network resource allocation, the algorithm firstly according to the needs of the business model calculated unloading problems based on server dormancy, then according to lyapunov optimization method based on energy consumption across time slot energy consumption model of queue, unmounting long-term energy consumption optimization problem into a time slot in the uninstall decision problem, Then the unloading problem was solved based on reinforcement learning.

Finally, change the setting parameters and simulation system and other algorithms, the experiment results show that the algorithm through the design according to the comprehensive cost of unloading task can be unloaded to the terminal, the edge server or processing on the cloud platform, can effectively save the energy consumption and high efficient distribution terminal equipment, edge of the platform, and cloud resources, meet the demand of network low latency.

Acknowledgement

Through the study of *Information Retrieval and Academic Writing*, I have learned how to retrieve the literature I need and how to write a paper well.

My teacher Mr. Xu teaches me a lot of academic guidance and paid much attention to my future work direction, introducing me a lot of technical development status. In the future, my goal will be clearer.

I hope I can engage in interdisciplinary research in the future and apply what I have learned to my life. I also hope that I can write a fluent and satisfactory paper and publish it in my undergraduate study period. I think that would be exciting and something to be proud of. I have a long way to go.

References

[1] 唐宁. 边缘计算的物联网任务处理策略研究[D].北京邮电大学

[2] 任晨珊. 边缘计算中的高能效与负载均衡技术研究[D].北京邮电大学,2019.doi:10.26969/d.cnki.gbydu.2019.000040.

[3] 王凯. 多接入边缘计算中的任务卸载决策研究[D].西安建筑科技大学,2021.

[4] 刘庆川. 面向能耗和时延的云边协同网络计算卸载和迁移算法[D].北京邮电大学,2021.doi:10.26969/d.cnki.gbydu.2021.001074.

[5] 王元君. 星地混合网络中的计算资源分配和负载均衡[D].北京邮电大学,2020.doi:10.26969/d.cnki.gbydu.2020.002021.

[6] 闫雨涵. 移动边缘计算场景下的资源分配与计算卸载策略研究[D].长春理工大学,2021.

[7] C. Qiu, H. Yao, F. R. Yu, F. Xu and C. Zhao, "Deep Q-Learning Aided Networking, Caching, and Computing Resources Allocation in Software-Defined Satellite-Terrestrial Networks," in *IEEE Transactions on Vehicular Technology*, vol. 68, no. 6, pp. 5871-5883, June 2019, doi: 10.1109/TVT.2019.2907682.

[8] Y. Tu, H. Chen, L. Yan, Offloading Decision Problems for Edge Computing in IoT Systems: Modeling， Solution and Classification, Journal of Chinese Computer Systems , 2021, 1000-1220(2021) 10-2145-08

[9] H. Chen, Y. Hu, Task Offloading Model Based on 5G Edge Cloud Collaborative Distributed Network Architecture, doi:10.3969/j.issn.1006-1010.2021.04.024

[10] G. Zhang, X. Chen, H. Xu, Resource Allocation Based on Unsupervised Deep Learning in Mobile Edge Computation Network, doi:10.13757/j.cnki.cn34-1328/n.2021.04.001

[11] Z. Cai, Z. Li, Research on Mobile Edge Computation Offloading Strategy for Industrial Internet of Things

[12] C. Li, J. Li, X. Xu, T. Li, Research on mobile edge computing resource allocation with energy harvesting device, doi:10.16182/j.issn1004731x.joss.21-0576

[13] Y. Xian, Q. Song, C. Guo, C. Liu, Method of Task Offloading and Resource Allocation in MEC with Limited Computing Resources

[14] B. Lei, Z. Liu, X. Wang, M. Yang, Y. Chen, Computing network: a new multi-access edge computing, doi:10.11959/j.issn.1000−0801.2019209

[15] H. Chang, J. Feng, C. Duan, C. Yan, K. Xia, Multi-Tier Transmission Control of Marine Observation Data, doi:10.15918/j.tbit1001-0645.2020.056

[16] Z. Zhang and S. Li, "A Survey of Computational Offloading in Mobile Cloud Computing," *2016 4th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*, 2016, pp. 81-82, doi: 10.1109/MobileCloud.2016.15.

[17] H. Jeong, I. Jeong, H. Lee and S. Moon, "Computation Offloading for Machine Learning Web Apps in the Edge Server Environment," *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, 2018, pp. 1492-1499, doi: 10.1109/ICDCS.2018.00154.

[18] C. Huang, Z. Wu and S. Lin, "The Mobile Edge Computing (MEC)-Based VANET Data Offloading Using the Staying-Time-Oriented k-Hop Away Offloading Agent," *2019 International Conference on Information Networking (ICOIN)*, 2019, pp. 357-362, doi: 10.1109/ICOIN.2019.8718188.

[19] J. Wang, T. Lv, P. Huang and P. T. Mathiopoulos, "Mobility-aware partial computation offloading in vehicular networks: A deep reinforcement learning based scheme," in *China Communications*, vol. 17, no. 10, pp. 31-49, Oct. 2020, doi: 10.23919/JCC.2020.10.003.

[20] E. Jonas, "Cloud Programming Simplified: A Berkeley View on Serverless Computing," arXiv:1902.03383.

[21] G. Cai and B. Yu, "Event-based awareness promotion for distributed collaborative activities," *2014 International Conference on Collaboration Technologies and Systems (CTS)*, 2014, pp. 302-309, doi: 10.1109/CTS.2014.6867581.

[22] F. Xhafa and A. Poulovassilis, "Requirements for Distributed Event-Based Awareness in P2P Groupware Systems," *2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops*, 2010, pp. 220-225, doi: 10.1109/WAINA.2010.112.

Junyao Hu, born in 2001, is a junior student of China University of Mining and Technology major in computer science. His main research interests include cloud computing, machine learning and software engineering technology.

Email: `06192081@cumt.edu.cn`

GitHub: `github.com/JunyaoHu`