Journal of Computers Vol. 32 No. 2, 2021, pp. 233-239
doi:10.3966/199115992021043202020

Design of Environmental Perception System Based on Edge Computing

Hongrui Zhang[1*], Qinghai Wang[1], Jianhong Feng[2], Zhaohong Yang[1], Hongyu Wang[1]

[1] Department of Information and Communication, Academy of Army Armored Forces, Beijing, China
RuiRuidabing@163.com

[2] Unit 32366 of the Chinese People's Liberation Army, Beijing, China

Received 2 January 2021; Revised 2 February 2021; Accepted 31 March 2021

Abstract. For the characteristics of marginal environment such as battlefields which environment is very difficult to measure, environmental perception has difficulty, resources are scarce, communication conditions are inferior and energy issues are highlighting. Depending on the issue that current architecture cannot process data in real time the edge calculation technique is proposed and aims to address the shortcomings of cloud computing in processing data that cannot be remedied. According to the EdgeX open source framework, an edge computing platform is designed in this paper. And realizes the three-layer network structure of "terminal edge-cloud" or "terminal-intelligent gateway-edge cloud-cloud". According to the design idea of loose coupling, the edge computing platform realizes the loose coupling between microservices and the separation of management and service. A resource scheduling strategy based on depth Q-learning is used to realize adaptive scheduling, which can effectively deal with the fluctuation of load demand and maximize the utilization of resources.

Keywords: Computing platform, Edge computing technology, Cloud, Q-learning, Resource Scheduling Strategy

1 Introduction

In the present age, the army relies on cloud computing, big data and other technologies to summarize the battlefield environment information in real time to the cloud, and the cloud relies on big data processing technology to send the processing results to the command center. However, the current architecture does not allow real-time data processing based on the current architecture. In addition, the battlefield situation is usually rapidly changing, and it should be taken into account that data acquisition in some scenarios is often temporarily interrupted by force majeure. Compared to cloud computing, Edge calculator is a small 'cloud' processing centre near the terminal equipments, that is providing computing processing and storage capacity at the edge of the network to ensure the timeliness of data processing [2].

2 Related Work

Domestic scholars Zhang Kaiyuan and GUI Xiaolin et al. carried out research on computing migration and content caching in mobile edge networks, 4 key research issues in MEC architecture, computational migration, edge caching, and service choreography were comprehensively studied and analyzed [1].

Domestic scholars Lei Bo, Liu Zengyi et al. proposed a computing power network scheme based on the deep fusion of cloud network edges to solve the problem of computing power collaboration between edge computing and cloud computing which can effectively deal with the multilevel deployment of future business and flexible scheduling among nodes at all levels. However, most of the current research remains in the level of theoretical and algorithm research, and the description of the environment

* Corresponding Author

awareness system based on edge computing is not detailed, so it cannot be implemented in a concrete way [8].

3 System Design

3.1 System Frame Construction Design

According to the requirements of edge computing and Internet of Things application scenarios, the edge computing platform can be functionally divided into modules as shown in the figure below: device management module, rule engine module, message routing module, scene service module, user management module and system management module, the system management module can be divided into monitoring module, operation and maintenance module and log module. Modules differentiated by functionality may manage multiple parts of the EdgeX at the core layer, support layer, device layer, and export layer simultaneously.

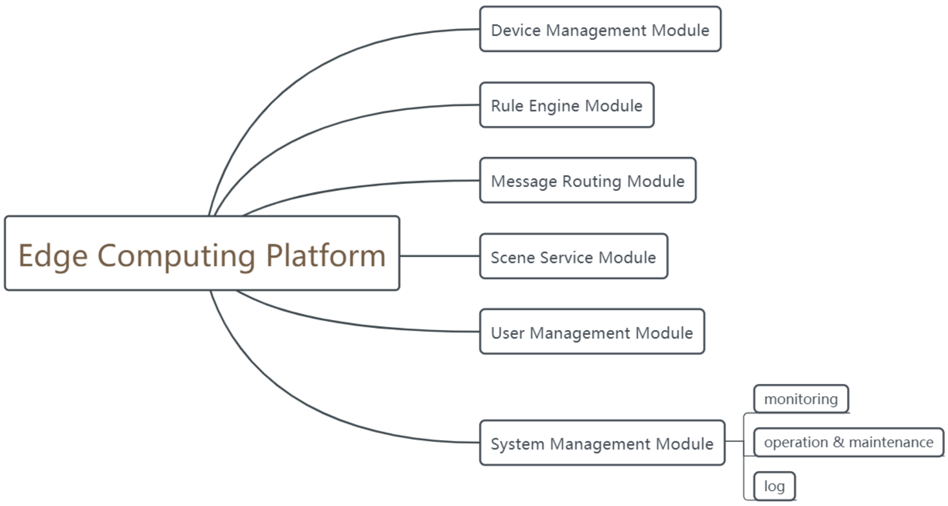

Fig. 1. Function module of edge computing platform

The functions of each module are described as follows:

(1) Device Management Module:

The device management module is responsible for defining, connecting, deleting, viewing and other operations of the device, additional support for individual protocols should also be developed in this module. The device management module should be divided into two micro services: device management and protocol support. The device management micro service is responsible for providing API such as new connection, deletion and viewing details of the device, protocol support micro services are responsible for the protocols used by the device, one protocol corresponds to one micro service. When a user sets up a gateway, a single protocol can be deployed separately according to the protocol of the device connected to the gateway to save storage and computing resources of the smart gateway. Device management realizes the separation of specific device and device model template, and the same model device adopts the form of calling template to simplify device Settings.

(2) Rule Engine Module:

The rule engine evolved from a reasoning engine and is a component embedded in an application that implements the separation of business decisions from application code and the writing of business decisions using predefined semantic modules. Using a rules engine can reduce the maintenance and extendable costs of an application by reducing the complexity of the components that implement complex business logic.

(3) Scene Service Module:

Scene service is composed of device service, which can realize user-defined scene, batch execution of device service, and realize encapsulation and combination of device functions. Device functions are defined by the device template in the device management module and provided by the specific device added. When the micro-service of other modules needs to use the device function, it sends the request to the scenario service module through message queue in the form of request to open the service, The device service is started and returned by the micro service of this module.

(4) Message Routing Module:

The message routing module provides the management of message queues, sets the forwarding rules between message queues, provides simple message processing functions, and is responsible for sending function calculation data into the function calculation module.

(5) User Management Module:

User management module achieves user registration, login and authority management, users are managed by the user group they are in, different user groups have different levels of permissions and the super administrator should be able to set permissions for user groups.

(6) System Management Module:

The system management module includes sub-modules such as monitoring operation and maintenance logs, which realizes the management of the edge computing platform system and is responsible for pushing notifications to users.

3.2 System Security Design

(1) Data center security assurance (Network platform security and part of the physical environment security)

Data center physical security main index according to the center of fire prevention and theft and other related work, to prevent the destruction and loss of equipment, the need to operate and maintain the host personnel identity authentication. In addition, the data center should also have the ability and facilities to provide the following network security measures, including: hardware firewall services, virus detection services, system vulnerability scanning and detection services, network attack detection services.

(2) Application platform security guarantee

a. Encryption Transmission

In order to prevent hackers from monitoring the network and data interception, the maintenance of the system is completed through SSL. In SSL, public key and private key are used for encryption, and public key is used during connection establishment, the type and strength of encryption is determined by the process of establishing the connection between the two ends, within each SSL session(both the client and the server were identified there), the server is required to complete one operation using the server private key and one operation using the client public key. All systems currently use RSA encryption, and one SSL session requires only one “tough” encryption operation

b. Prohibit downloading unconfigured files

The system reads only files, performs separation in order to prevent program browser users from downloading program files. And the system optimizes web server to avoid exposure of program code.

c. Registration verification

In order to prevent unregistered users from directly entering the system without registering the interface, using Session object for registration verification. Manage and maintain system Session to enforce users to carry out system security authentication

d. Verify the validity of the access connection

Another measure to verify the validity of access connection is to detect the source of system access. For access that is not verified by the system and generated by the server, the system issues a warning and denies access. This method effectively avoids illegal access via URL

e. Security of three-tier structure

The system adopts a three-tier structure, all business logic is completed by the business logic layer, and this module built-in control of access security, this structure adds another layer of security to the system. In addition, the business logic layer encapsulates the way and mechanism of accessing the database and is responsible for data access. Terminal browser users can’t touch the access mechanism to the database which effectively avoids the hidden danger of data security caused by traditional page accessing data directly and accessing data through CGI.

f. Specifies the default page for all virtual directories

(3) Local maintenance environment safety guarantee

There are two aspects to maintaining environmental security: local physical and network environments’ security. The security of the local physical environment refers to the security of the computer room and equipment for the remote maintenance of the website, including the maintenance of terminals, local servers and local networks. In daily operation, the website maintenance department is responsible for the security part.

4 Key Technology Research

4.1 Resource Scheduling Strategy Based on Deep Q-learning

In edge computing resource scheduling system, resources change all the time, and there is no prior information available on the system state. The scheduler must learn when it runs, that is, there is no ‘correct’ result to learn. Reinforcement learning, on the other hand, is not limited to prebuilt models. It improves behavior by interacting with the environment to evaluate the ‘great’ rating of current behavior. In other words, agents in reinforcement learning use feedback-based scheduling mechanism to observe and update the environment in each learning stage, so that the scheduling results are closer and closer to the optimal solution. Thus, reinforcement learning is more suitable for the adaptive scheduling of edge computing resources [6].

When the task arrives in the scheduling system, the whole scheduling process is shown in Figure 2. First, the tasks are sorted according to the task type, the task deadline and the task wait time, and the dynamic priority is designed to complete the preprocessing of the task. Then, according to the CPU utilization, resource availability, load change characteristics, energy consumption characteristics and other state information in the system, the response time and resource utilization ratio of the system were analyzed, and the tasks were assigned to specific virtual machines using the feedback mechanism adaptive scheduling strategy provided by reinforcement learning [4]. The process is as follows: Action A is the decision of resource allocation, which is mainly to select the appropriate virtual machine for the task; State S is the use of resources, and the feedback mechanism R is comprehensively determined by such indicators as the utilization rate of system resources and the execution time of tasks. It mainly generates feedback (reward or punishment) for each pair of state-action.

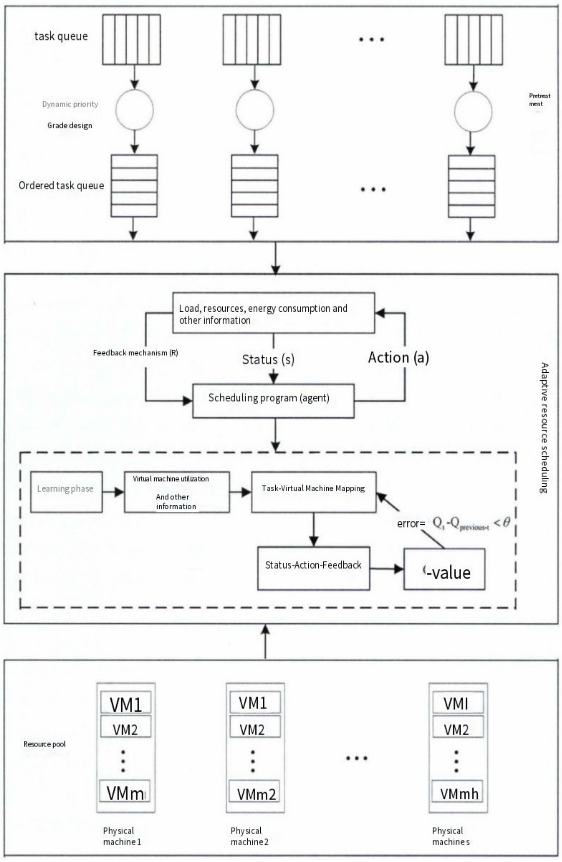

Fig. 2. Adaptive resource scheduling mechanism

In the scheduling process, the resources selected will be different if the execution order of tasks is different, and the required completion time, including the waiting time of tasks and the execution time of tasks will be affected, thus affecting the process of the entire scheduling process [7]. Therefore, in order to effectively avoid the blindness of initial action selection in the course of reinforcement learning and training, improve the convergence speed. In this paper, before the task selection of virtual machine, it is necessary to preprocess the task, design dynamic priority and sort the task, which is beneficial to improve the efficiency of resource scheduling policy.

According to the characteristics of edge computing resource scheduling problem, this paper adopts q-learning algorithm based on value function in reinforcement learning, which learning process is that the scheduler explores the value function with cumulative evaluation according to the feedback value of the state-action pair. The reasons for using q-learning algorithm in this paper mainly include :

(1) The algorithms based on value function mainly include Temporal difference(TD) and Monte Carlo (MC). In the resource scheduling environment, MC does not update knowledge until the end of the scheduling, and if the scheduling process is too long, a large update delay will occur, TD is updated immediately after a time step, constantly iterating through resource allocation, so TD and its improved methods are widely used. TD can be divided into online strategy Q-Learning algorithm and offline strategy SARSA algorithm according to the choice of strategy, in resource scheduling problems, there is uncertainty in resources, so the q-learning method is more suitable for the diversity of data changes;

(2) As a complex NP-hard problem, resource scheduling requires the continuous declaration of parameters, but using too many parameters will bring a lot of computational overhead to the system, especially the AC algorithm [3]. Among all the methods of reinforcement Learning, only q-learning algorithm describes the problem through the pair of state-action parameters, so it is more suitable for the research direction of this topic;

(3) Policy-based algorithms need to be trained with the same strategy, but this method is not suitable for edge computing resource scheduling because of the high load volatility in edge computing environment. In addition, the uncertainty of resources in edge computing environment causes much feedback, and it is not suitable to use the method based on strategy iteration [5].

The Q-Learning algorithm was proposed by Watkins in 1989. The ultimate goal of this algorithm is to achieve the optimal strategy. Each state-behavior pair of this algorithm corresponds to a desired cumulative reward called the value of Q, namely the value function. According to the definition, it can be seen that the definition of value function recognition is based on continuous time. The Q-value correction formula of single-step Q-learning is shown as follows:

$$Q^{(t+1)}(s_1, a_t) = Q^{(t)}(s_1, a_t) + a[r + \gamma \max Q^{(1)}(s', a') - Q^{(t)}(s_t, a_t)] \quad \textbf{(1)}$$

Where, $\alpha \leq 1$ is the learning rate, and $0 < \gamma < 1$ is the penalty term. The choice to help the function converge is related to the convergence rate, and also has an impact on the final result. If the value of γ is too small, it will have little influence on the reward and punishment of subsequent states, and it is not easy to get the optimal solution, and it is easy to get the suboptimal solution. However, if the choice of γ is too large, the convergence speed of the algorithm will slow down. In the right-hand side $Q^{(t)}(s_t, a_t)$, the upper index t refers to the number of iterations, which is the old Q value, and is the estimated value of the subsequent states. In each step of the training process, the current state s_t selects action a_t to enter the next state s', and updates the current state with the combination of the immediate reward and punishment value r at that time and the estimated value of the value function of the subsequent state obtained by greedy algorithm. The calculation method of the whole process is shown in algorithm 1.

During each scheduling selection action decision, the agent selects the action using some strategy (for example, ϵ-greedy strategy), and by constantly updating the state space, the agent will converge faster toward the optimal direction, accelerating the convergence speed, and making its value function (i.e., the Q value) approach the optimal direction continuously.

Algorithm 1. Q-learning

1. Initialize
2. for each episode do
3. Select a status in S

for each time of episode do
  according to s_t and $Q(s, a)$, use strategy to select action a_t
  execute action a_t, get the next status and feedback value r
  $Q^{(t+1)}(s_1, a_t) = Q^{(t)}(s_1, a_t) + a[r + \gamma \max Q^{(1)}(s', a') - Q^{(t)}(s_t, a_t)]$
   $s_t \leftarrow s'$
end for
end for
Until s is the goal status

In the edge of the computing resource scheduling system, resources would not make a user a task, at the same time it is impossible to only one task requires service, local decision-making optimization does not mean that the efficiency of the system as a whole is highest, on the contrary, there is the behavior of the possible state of subsequent actions resulting in a loss of performance so that the overall average efficiency reach optimum, so we adopt the Q - Learning is more suitable for the adaptive scheduling algorithm. The solution to the optimal strategy for resource scheduling depends largely on the precise definition of state, action, and reward and punishment functions. In this report, are respectively defined as:

(1) State Space (S)

State can be defined as the state of each virtual machine performing a task and can be represented as a vector. For example, at time T, $s_t = (1, 0, \dots, 1)$ indicates that the first virtual machine is currently occupied by a task, the second virtual machine is idle, and the last virtual machine is occupied by a task

(2) Motion Space (A)

For the nth task request, we define its action space as $(0/1)_m^n$, which means whether the nth task request is provided to the mth virtual machine for execution. For example, $a_m = (0, 1, 0, \dots, 0)$ means that the nth task request is assigned to the second virtual machine at time t.

(3) Rewards and punishments function. (r)

The reward and punishment function is used to reflect the correct running state and the scheduling efficiency of the system. With the development of edge computing, more and more computing and storage resources are concentrated in the edge resource pool, resulting in a sharp increase in energy consumption and related costs. The problem of high energy consumption can not only cause the waste of electricity, but also lead to the deterioration of environmental quality, and also affect the operating efficiency of the system. Therefore, energy consumption has become an important measure of resource scheduling efficiency. Research results show that the resource utilization rate of computing nodes has a significant impact on energy consumption, and energy consumption generally changes with the change of CPU utilization, when the resource utilization rate of the computing node is very low, there is still 50% energy consumption. However, the higher the resource utilization rate is, the slower the energy consumption power growth will be. Therefore, from the perspective of improving resource utilization and reducing energy consumption, we designed the reward and punishment function by reducing the waiting time of tasks under the constraint of cut-off time, the details are expressed by the following formula:

$$r_1 = maximum \left\{ \frac{\sum_{i_{local}}^{j_{local}} u_{i,j}}{i_{local}} \right\} \&\& maximum \left\{ \frac{1}{overw_j} \right\}, subject\ to\ w_j < SLA_{deadline} \quad \textbf{(2)}$$

Including i, as a physical machine i, the number of the virtual machine by i_{local}, said j to i physical machine a virtual machine $u_{I,j}$, i is the case of a physical machine I j virtual machine utilization, which can be a physical machine i all virtual machine utilization to an average of $\sum_{j=0}^{i\_local} u_{i,j}$, $averw_j$ said task the average waiting time in the virtual machine j, w_n is the deadline for the current task n. For a given the current task, if the task is assigned to a virtual machine, the virtual machine is in the physical machine utilization rate of the average score of the average utilization ration of other virtual machine, at the same time the task assigned by the virtual machine the wait time is less than the original waiting time, and meet the SLA or QoS constraints, the scheduler will receive positive incentives, the value is 1; If the target function is not met and the response time of the task violates SLA or QoS constraint, it will be punished with a value of -1; Otherwise 0 [4].

5 Future Work

The explosive growth of mobile Internet application will be put forward for the distribution of the existing network resources are extremely stringent requirements, according to the application of the mobility characteristics of resource demand uncertainty, intelligent management section is needed to improve resource utilization, in order to improve the network quality and user experience, as a new generation of Internet of things application based services and provide strong technical support. A low delay mobile Internet of Things (IOT) slicing model based on multi-edge collaboration is established to meet the dynamic demand of IOT terminals for slicing resources during the mobile process. By combining multiple edge nodes, the model establishes a distributed deep learning platform for multi-edge node collaboration, and the formation of multi-edge collaboration model is the future research direction of this paper.

References

[1] K.-Y. Zhang, X. L. Gui, D.-W. Ren, J. Li, J. Wu, D.-S. Ren, Survey on Computation Offloading and Content Caching in Mobile Edge Networks, Journal of Software 30(8)(2019) 2491-2516.

[2] L. Z. Li, P.-L. Zhou, P. Cheng, Z.-G. Shi, Architecture, challenges and applications of edge computing, Big Data Research (2)(2019) 3-16.

[3] M. Deng, H. Tian, B. Fan, Fine-granularity based application offloading policy in small cell cloud-enhanced networks, In: Proc. of the IEEE ICC. IEEE, 2016, 638-643.

[4] M. Satyanarayanan, P. Bahl, N. Davies, The case for VM-based cloudlets in mobile computing, IEEE Pervasive Computing 8(4)(2009) 14-23.

[5] J. Zhu, D.-S. Chan, M.-S. Prabhu, P. Natarajan, Improving Web sites performance using edge servers in fog computing architecture, In: Proc. of the IEEE 7th Int'l Symp. on Service-oriented System Engineering, IEEE Computer Society, 2013, 320-323.

[6] Y. Mao, J. Zhang, K.-B. Letaief, Dynamic computation offloading for mobile-edge computing with energy harvesting devices, IEEE Journal on Selected Areas in Communications 34(12)(2016) 3590-3605.

[7] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, Y. Zhang, Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks, IEEE Access 4(99)(2017) 5896-5907.

[8] B. Lei, Z.-Y. Liu, X.-L. Wang, M.-C. Yang, Y.-G. Chen, Computing network: a new multi-access edge computing, Telecommunications science 35(9)(2019) 44-51.