Intelligent Universal Acceleration Framework and Verification for Edge Cloud Applications

Jie Mei[†], Bo Lei*, Xuliang Wang*, Xing Zhang[†], Qianying Zhao*
[†]Beijing University of Post and Telecommunications, Beijing, P.R. China
*Beijing Research Institute of China Telcom, Beijing, P.R. China

***Abstract*—With the development of 5G, the applications impose more diverse and higher performance demands on the infrastructure capability of Edge Cloud. In order to meet the demands, many hardware-based Edge Cloud Application Acceleration schemes have been proposed. However, to ensure the vigorous development of Edge Cloud applications, how to efficiently match and schedule the acceleration capability of edge cloud applications with the diverse acceleration demands of edge cloud applications is one of the indispensable prerequisites. In this paper, an intelligent general acceleration framework for Edge Cloud applications is proposed, which mainly solves the following problems: first, distinguish and model the acceleration demands of applications accurately; second, abstractly model and classify the general acceleration capabilities; third, detect the initial period of application acceleration, and provide intelligent policy scheduling framework according to test results. And the framework achieves accurate matching and scheduling between application acceleration demands and general acceleration capabilities. So, the Intelligent Universal Acceleration Framework can accelerate the application of Edge Cloud accurately, and ensure the efficient utilization of resources. In addition, bandwidth-intensive, face recognition and enterprise VPN gateway are used to validates that the Intelligent Universal Acceleration Framework is helpful to analyze, match and schedule the acceleration demands of applications and the general acceleration capabilities of infrastructure.**

Keywords—MEC, edge computing, application acceleration

I. INTRODUCTION

With the rapid development of 5G and industrial Internet, in many new vertical industries, the demand for Edge Computing is mainly reflected in three aspects: delay, bandwidth and security. At present, the demand for Edge Computing in vertical areas is the clearest, such as smart cities, smart manufacturing, and enterprise dedicated lines. For smart city, based on locally deployed Artificial Intelligence (AI) acceleration servers, Edge Computing can realize intelligent image analysis in milliseconds, such as face recognition and object recognition. For smart manufacturing, by deploying forwarding acceleration servers in Edge Cloud, it is possible to effectively accelerate the fast forwarding of message traffic between multiple machines in smart manufacturing, while reducing the delay of communication between machines; and for enterprise dedicated lines, deployment of accelerated servers can greatly improve the concurrent processing capabilities of enterprise users' SSL (Security Socket Layer) VPN (Virtual Private Network) (mobile terminal access scenario) and IPSEC (Internet Protocol Security) VPN (fixed network access scenario), and provides secure connection services for enterprise dedicated line users in centralized, intensive and efficient way [1]. However, there are many acceleration demands in the above business. That is to say, how to achieve efficient scheduling of resource and optimization of the whole system become the key to research, after evaluating the diversified acceleration needs scientifically and objectively. Therefore, an Intelligent Universal Acceleration Framework for Edge Cloud Applications emerges as the times require .

II. RELATED WORK

Based on the idea of "promoting the marginalization of cloud computing", the industry has carried out extensive and in-depth research on the application framework of Edge Computing from different perspectives.

KubeEdge, a kubernetes-based computing framework for Cloud Computing, can achieve cloud-side collaboration, computing power sinking, mass device access and other functions relying on the container layout and scheduling capabilities provided by Kubernetes itself. It supports the arrangement and management of edge clusters, and provides powerful support for the rapid and effective extension of Cloud Computing capabilities to Edge Computing [2]. KubeEdge customize and optimize runtime in depth, so that K8S-based applications can run at the edge. A reliable message channel is designed and implemented to deal with the unstable factors of edge network, which simplifies the access complexity of equipment and greatly reduces the cost of configuration and operation. However, KubeEdge is only a software platform for edge nodes. There are still many scenarios and functions that enterprise users need to improve urgently. It also needs the cooperation of cloud management platform to build a complete production and management platform for edge computing.

OpenEdge is a self-developed Edge Computing framework of Baidu Cloud. It is mainly designed to fit the application of industrial Internet scenarios, expand computing capabilities to user sites, and provide users with temporary offline, low latency computing services. The computing services of OpenEdge include device connection, message routing, remote synchronization, functional computing, video access preprocessing, AI reasoning and so on. On the one hand, OpenEdge implements modularization and decomposes the main functions to ensure that each sub-function module operates independently and without influence, which can fully meet the practical demands of users'on-demand use and on-demand deployment; on the other hand, OpenEdge also adopts the idea of comprehensive container design. Based on the mirror of each module, it can be constructed one-click on various operating systems supported by Docker. By utilizing the resource isolation capability endowed by Docker container mode, it can accurately allocate CPU(Central Processing Unit), memory and other resources of each running instance and improve the efficiency of resource utilization. However, OpenEdge only supports the most common service scenarios of Edge Computing currently, such as localized image recognition, target detection, fault diagnosis and so on. And for how to support the growing diversity needs, OpenEdge gives no solution.

EdgeX Foundry is an open framework based on IoT(Internet of things) Edge Computing created by the Linux Foundation. It aims to simplify the process of developing and deploying solutions for industry, enterprises and consumers [3]. The framework has done the following work to standardize the edge computing architecture in the industrial Internet of Things: supporting any combination of device interfaces, standardizing connection protocols, connecting different IoT terminal software and firmware to the cloud, supporting heterogeneous components with general interoperability basis to provide flexible micro-service shelves for third-party products, and allowing functional modules to be distributed among multiple edge hardware nodes or processors of a given node to achieve rapid establishment of networking devices from different vendors. However, EdgeX does not provide a solution for large-scale deployment.

The related researches above have made a lot of attempts in promoting mature Cloud Computing capabilities to Edge Computing of user applications, such as offline, low latency services and standardization of industrial IOT. However, the current Edge Computing frameworks are tightly coupled with specific application scenarios, and are not involved in such aspects as low latency, large bandwidth acceleration capability and resource scheduling, which are urgently needed by edge computing applications.

III. Design and Implementation

A. *Intelligent Accelerated Resource Architecture for Edge Cloud*

Aiming at the unified management of accelerating resource in current SDN/NFV(Software Defined Network/ Network Function Virtualization) architecture, an Intelligent Accelerated Resource Architecture for Edge Cloud is proposed in this paper, and the overall structure is shown in Fig. 1 From top to bottom are application intelligent scheduling layer, cloud tube layer, virtualization layer and heterogeneous computing hardware layer.

Among them, application intelligent scheduling layer includes upper application, heterogeneous demand analyzer and heterogeneous resource scheduler, where heterogeneous demand analyzer mainly differentiates various service scenarios, and heterogeneous resource scheduler intelligently matches various refined demands with virtualized resources to achieve accurate scheduling.

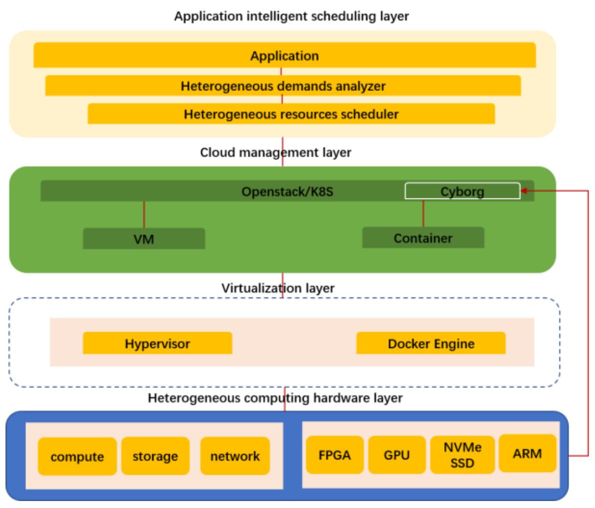

Fig. 1 Intelligent Accelerated Resource Architecture for Edge Cloud

Cloud management layer includes OpenStack/K8S cloud operating system, virtual machine, container and so on. And it can realize the virtualization of resources.

The virtualization layer includes Hypervisor and Docker Engine. And it can coordinate the access of the underlying computing hardware resources and provide protection between the upper virtual machines or containers.

Heterogeneous computing hardware layer mainly includes traditional X86 computing resources (computing, storage, network) and all kinds of acceleration resources (FPGA(Field-Programmable Gate Array), GPU, NVMe SSD, ARM). All kinds of resources mentioned above provide acceleration capability for all kinds of services through collaboration.

B. *Multidimensiona Refined Demand Model*

1) Key Service Scenarios for Edge Clouds: MEC is mainly used in scenarios with high real-time demand, large amount of data, high processing capability and strong localization demand.

a) Mobile Video: With mobile video traffic explosively increasing, network delay will reduce the perception of mobile video audiences. So mobile video stagnation and buffering is still a big problem for operators and their customers [4]. MEC is the key solution to solve this problem. MEC server is close to the wireless side, and content can be cached to the MEC server in advance. When there is a need to watch mobile video, the MEC server immediately checks whether there is content requested by the user in the local cache. If there it is, MEC provides directly; if not, MEC goes to the network service provider to get resource and caches it locally. And when other users have the same demand next time, MEC can provide services directly, which not only reduces the request time, but also solves the problem of network congestion.

b) Vehicle Networking: Vehicle networking has two demands: low latency and high reliability. And with the increase of networked vehicles and the explosion of data volume, the challenges to delay and reliability are also higher. After the application of MEC, MEC server can accurately and real-time perceive the change of vehicle position to ensure the reliability of communication; the data of vehicle network can be stored in the server near the vehicle, which can greatly reduce the end-to-end delay of the existing network and send auxiliary driving information such as alarm to OBU (vehicle-mounted unit) in emergencies to achieve emergency operations such as emergency braking.

c) AR (Augmented Reality): AR applies virtual information to the real world. It can improve people's visual experience when superimposing AR content on the field of view taken by camera. The high delay will make the viewer dizzy. The real-time processing of information transmitted by MEC through AR devices can greatly reduce the delay, improve the accuracy of data processing and enhance the user's feeling. What's more, there is no need to put too high demand on the memory and storage capacity of AR devices.

d) Monitoring Video: At present, there are two common methods of data processing of surveillance video:

camera processing and server processing. For camera processing, each camera requires data analysis capabilities, making the cost expensive. For server processing, amount of data needs to be uploaded to the server, which will increase the burden of the core network, with large delay and low efficiency. By deploying video management applications on MEC servers, however, monitoring video data can be localized without uploading a large number of video data to the server, reducing the burden of the core network and improving efficiency. Moreover, the solution does not require the camera to have data analysis capabilities and reduces the cost.

2) Service-Demand Interaction Model of Edge Cloud: For the different types of service described in the last section, the delay demands are shown in Table I below. The lower delay demand, the higher the service rate demand for virtual machine. For different servers and deployment locations of them, the service rates and transmission rates are shown in Table II below.

TABLE I Different types of application parameters

Types of application	Delay demands (ms)	Corresponding Virtual Machine to Service Rate
Autopilot signaling	1	1500
AR	2	800
Health monitoring	5	500
Online Video	10	400
Virtual Reality（VR)	20	300
Online games	30	200
Video Service	50	100

TABLE II Server and network parameters

Parameters	Values
Cloud Server CPU	1×10^{10}cycles/s
Edge Server CPU	5×10^{10}cycles/s
Mobile device to edge transmission rate	72.2Mbps
Transport rate from edge to cloud	15Mbps
Edge Server Resource Quantity	200MB

From the two tables above, we can see that we should design a refined multi-dimensional demand model, and refine the service scenario into service demands. And through the call of cyborg accelerator card, we can accelerate the capacity improvement and better achieve resource scheduling. Modeling based on service-demand issues as described below.

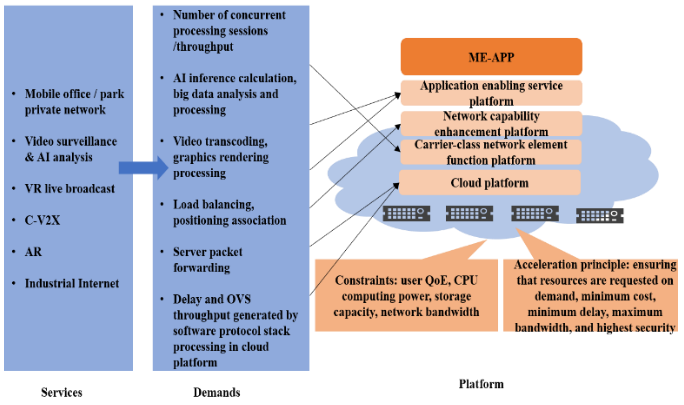
Mobile office / park private network
Video surveillance & AI analysis
VR live broadcast
C-V2X
AR
Industrial Internet
Number of concurrent processing sessions /throughput
AI inference calculation, big data analysis and processing
Video transcoding, graphics rendering processing
Load balancing, positioning association
Server packet forwarding
Delay and OVS throughput generated by software protocol stack processing in cloud platform
ME-APP
Application enabling service platform
Network capability enhancement platform
Carrier-class network element function platform
Cloud platform
Constraints: user QoE, CPU computing power, storage capacity, network bandwidth
Acceleration principle: ensuring that resources are requested on demand, minimum cost, minimum delay, maximum bandwidth, and highest security
Services
Demands
Platform

Fig. 2 Service-Demand Interaction Model

Service-demand interaction model is shown in Fig. 2. The platform architecture based on Edge Cloud service scenario is mainly divided into cloud platform, telecommunication-level network element function platform and network capability enhancement platform.

In this graph, the refined demand model of Edge Computing extracts independent service sub-demands from several key service scenarios of edge computing, and carries out fine analysis of service sub-demands.

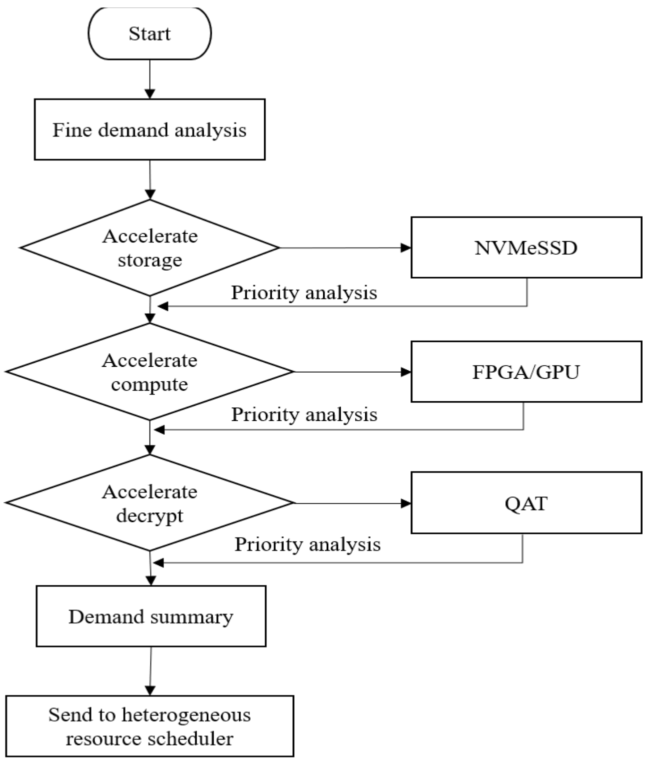

Fig. 3 Flow chart of refined demand model

3) Flow chart of refined demand model: The refinement demands model process includes the following steps as shown in Fig. 3. Firstly, analyze the refined demand analysis for service scenarios, and judge whether the demand needs to accelerate storage according to the analysis. If necessary, invoke NVMe SSD through Cyborg to accelerate hardware-related performance parameters. Secondly, analyze whether the demand needs to accelerate calculation. And if necessary, invoke related performance parameters of accelerated hardware, such as FPGA/GPU. And then judge whether it is necessary to speed up decryption. If necessary, select the performance parameters of QAT accelerated hardware. After all the judgments of demand analysis are completed, aggregate each demand and send the summary to heterogeneous resource scheduler.

4) Fine Demand Analysis Algorithms: In this section, a refined demand analysis algorithm is proposed, which deploys the heterogeneous hardware required by the service- demand model in different categories, and match appropriate weights for each dimension to better reflect the resource demands.

The idea of algorithm design is as follows: firstly, the refined demand analyzer builds the demand analysis model based on CPU, memory, network bandwidth, disk IO, accelerated resources (GPU, FPGA, ARM, NVMe SSD); secondly, choose appropriate weights and set the weight priority of each demand; thirdly, aggregate the calculated real-time comprehensive demands, and send the demands to heterogeneous resource scheduler.

The algorithm consists of three parts: 1) Establish data model. Collect utilization rates needed by the demand model, such as utilization rates of CPU, memory, network bandwidth, disk IO, accelerated resource, and so on. Based on the data, build a refined multi-dimensional demand performance model. 2) Assign weights. The weights range from 0 to 1, and the sum of them is 1. By default, the weights are CPU > memory > disk IO > network bandwidth > accelerated resources. And the priority can be updated in real time according to the demand data input from the service side to the demand model or the resource utilization rate of heterogeneous resource feedback. 3) Refine multi-dimensional sub-demands and obtain the weighted sum. And then send the aggregate to the heterogeneous resource scheduler [5].

1) Establish Model: Establish a model for refined multidimensional demand analysis. And performance model parameters include: CPU frequency (D_c), memory size (D_m), disk IO (D_{IO}), network bandwidth (D_n), accelerated resources (D_a). So the refined multi-dimensional demand model matrix can be described as:

$$D = [D_c, D_m, D_{IO}, D_n, D_a] \tag{1}$$

2) Assign weights: The larger the weight (k) value is, the higher the priority of resource invocation will be. The smaller the weight (k) value is, the lower the priority of demand will be, and the smaller the scope of revision will be. And each weight parameter k has different proportions in different service applications. For FTP, services need a large proportion of network bandwidth and disk IO; for Web services, it needs a large proportion of CPU and memory; for Edge Cloud-oriented services, the demands for CPU, network bandwidth, disk IO and acceleration resources is large because of relatively weak CPU processing capacity. In this case, CPU only assists in the allocation of services to the FPGA/GPU accelerator card or for preprocessing. And FPGA/GPU carries on the main calculation with strong performance.

3) Comprehensive demand analysis and calculation: Assuming that the weights of CPU, memory, disk IO, network bandwidth and acceleration resource are Ki, and the sum of the weights of the five parameters is 1. That is, K_1; K_2 K_3; K_4; K_5 correspond to the design CPU, memory, disk IO, network bandwidth, acceleration resource weights K_C; K_m; K_{IO}; K_n; K_a respectively. So, for service S_i, the comprehensive demand analysis formula is:

$$\begin{aligned} D(S_i) = k_c \times D_C(S_i) + k_m \times D_m(S_i) \\ + K_{IO} \times D_{IO}(S_i) \\ + k_n \times D_n(S_i) \\ + k_a \times D_a(S_i) \end{aligned} \tag{2}$$

$D_{(Si)}$ reflects the comprehensive demand of the service side. The larger the value of K_{Si}, the higher priority state the demand in, which we should take the lead in allocating virtualization resources. And when K_{Si} value is small, it means that the demand is in a low priority state. We can delay the allocation of virtualization resources to the demand. Weighting parameters of each demand can describe the service side demand to heterogeneous resources in detail, realize the optimal matching and maximum utilization of service demand and heterogeneous resources.

C. *Cyborg Component Architecture and the Process of Calling Accelerator*

With the development of 5G, Internet, Internet of Things, the demand for accelerating and uninstalling computing services is increasing day by day. Thus Cyborg project was born.

By managing and using accelerator hardware on computing nodes, Cyborg can provide various acceleration services for telecom operators in NFV and Edge Computing scenarios, improve user experience and reduce CPU load. The general hardware acceleration framework based on Cyborg is helpful for us to shield the differences of various acceleration hardware underlying implementations by using unified abstract logic and interactive system, and to standardize the processing capability of different hardware services. Operations and maintenance personnel can list, identify and discover accelerators by Cyborg, thus mount and unload accelerator instances, and install and unload drivers. The process of Cyborg calling the accelerator as shown in Fig. 4.

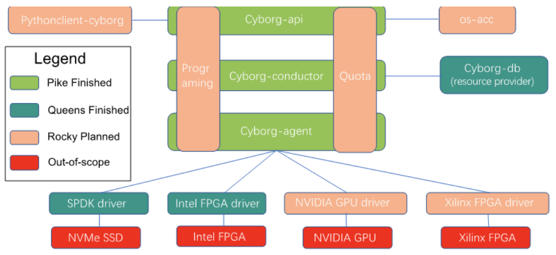

Fig. 4 Cyborg Call Accelerate Flow

Among them, the Cyborg API supports the basic operation of the accelerator, including attaching/detaching existing physical accelerators, listing additional accelerators and modifying accelerator status interfaces. Cyborg agent exists on computing hosts and other hosts that may use accelerators. The specific functions of agents include checking hardware to find accelerators, connecting instances to accelerators, sending data on available accelerators, status and utilization to Cyborg servers, scanning instance accelerators and using levels of existing accelerators, and so on.

Cyborg-Conductor-Cyborg-db database query and update operations need to be achieved by sending RPC requests to Cyborg-conductor service. Conductor is responsible for database access control and avoids direct access to the database.

The main functions of Cyborg-generic-driver include identifying and discovering additional accelerator backend, attaching accelerator to general-purpose backend, etc.

Although Cyborg has provided structure for most accelerated hardware to be called by virtualization layer, it still need improvements in intelligent scheduling of heterogeneous resources. If the resource scheduling algorithm can be improved on the basis of Cyborg hardware acceleration framework, the improved algorithm can effectively balance the load of nodes will, effectively improve the application performance of nodes, and enhance the user experience of the system.

D. Resource Scheduling Based on Cloud Tube Platform

This section mainly studies how to use the service-demand refinement model and resource scheduling model to analyze the existing resources, so that the demand and resources can be closely scheduled. And it can achieve the exact matching and scheduling between application acceleration demands and general acceleration capabilities Under the condition of unsatisfactory human matching effect.

1) Intelligent Heterogeneous Resource Scheduling Algorithm: In this section, based on OpenStack-Cyborg hardware acceleration framework, an improved intelligent strategy scheduling algorithm is proposed. And it can achieve accurate acceleration of various edge cloud applications, to ensure efficient utilization of resources.

The purpose of resource scheduling algorithm is to select a resource applicant to allocate resources. In a system with multiple resource applicants, the goal is to allocate resources according to the amount of resources that the applicant currently occupies and the resource quota allocated to it beforehand. DRF (Dominant Resource Fairness), the most widely used scheduling algorithm at present, is developed on the basis of Max-Min Fairness algorithm. Max-Min Fairness is the most widely used algorithm in one-dimensional resource scheduling, and it seeks fairness in resource allocation. Each time the allocation of resources takes into account the ratio of the resources (the applicant currently occupies to the total demand), and allocates the resources to the applicant with smallest proportion of resources [6]. DRF algorithm simplifies multi-dimensional resource scheduling into one-dimensional resource scheduling through the concept of main resource, which is the resource with the largest ratio of current usage to its quota.

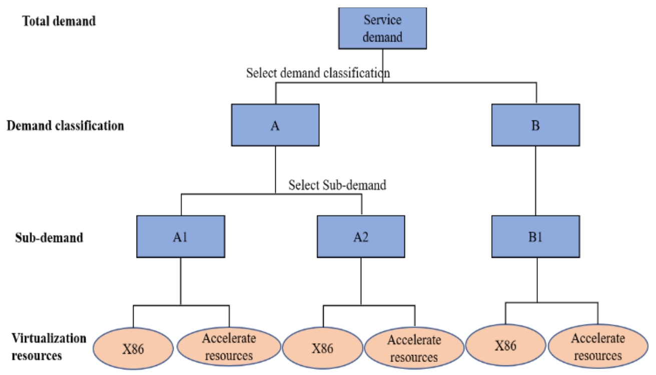

Fig. 5 Resource Scheduling Flow Chart

Fig. 5 illustrates the resource scheduling process of heterogeneous resource schedulers. Assume that the new cluster deployed is configured as follows: there are 36 virtual CPUs, 18 GB virtual memory, 24 virtual FPGA accelerators and 12 GPU virtual accelerators in the deployed new cluster. And for the sake of simplicity, the number of other accelerators in the resource pool to be 0. Demand A and B account for 1/3 and 2/3 of the total service resources, respectively. The Sub-demands subordinate to A occupies 4 virtual CPUs, 4 GB virtual memory, 2 virtual FPGA accelerators and 0 GPU accelerators. The Sub-demands subordinate to demand B uses 6 virtual CPUs, 8 GB virtual memory and 12 virtual FPGAs, and 2 virtual GPU accelerators. And the resource quotas for the two queues and the amount of resources already used are shown in Table III and Table IV.

DRF algorithm always allocates resources to the demand with small share of main resources. Under the condition of multi-dimensional resources, the share of a resource occupied in demand refers to the ratio of the amount of the resource occupied to the quota of the resource in demand. For example, the share of CPU resources in demand A is (4CPU)/(12CPU) = 1/3, and the share of each resource in the two demands is shown in Table V. So, the main resource of demand refers to the resource with the largest share of resources, i.e. the main resource (resource scheduling parameter) i is:

TABLE III Resource quotas allocated to the first-tier sub-queues

Demand	Resource proportion	CPU Resources	Memory Resources	FPGA Resources	GPU Resources
A	1/3	12	6	8	4
B	2/3	24	12	16	8

TABLE IV Number of resources used in the first layer subqueue

Demand	CPU Resources	Memory Resources	FPGA Resources	GPU Resources
A	4	4	2	0
B	6	8	12	2

$$i = \max_{i\ in\ resources} \frac{Used_i}{Demand_i} \tag{3}$$

TABLE V The proportion of the number of resources used to the resource quota in the first level demand

Demand	CPU Resources	Memory Resources	FPGA Resources	GPU Resources
A	4/12	4/6	2/8	0
B	6/24	8/12	12/16	2/8

Among them, $resources$ are a collection of resource types in heterogeneous resource platforms, and $Demand_i$ is the number of resources i that have been used by a demand. $Used_i$ is the quota of the demand on resource i, which is matched according to the refined multi-dimensional demand model. According to the formula above, the main resource of demand A is memory, and the main resource of demand B is the accelerator core of FPGA. According to the idea of DRF algorithm, the main resource share of A is 4/6, less than 12/16 of B. So we should allocate resources for A in the resource scheduling of the first level. The same method is used when deciding to allocate resources for sub-demands A1 and A2 of A, and recursively layer by layer, until the lowest demand at the demand level.

2) Heterogeneous Resource Scheduling Model: According to the above intelligent heterogeneous resource scheduling algorithm, the resource scheduling parameters can be obtained. In this section, a heterogeneous resource scheduling model is proposed, which shows that the heterogeneous resource scheduler cooperates with Cyber, to intelligently schedule the underlying resource flow. Firstly, fine multi-dimensional demand analyzer sends comprehensive demand information to heterogeneous resource scheduler. After that, heterogeneous resource scheduler sends real-time update scheduling parameters to Cyborg API interface. Then Cyborg Agent in heterogeneous resources sends real-time resource

status utilization rate to Cyborg server. And Cyborg server sends task completion status to Cyborg API. After that, API interface executes attach/detach heterogeneous resources according to scheduling parameters of heterogeneous resources and task completion. Finally, server sends real-time status of resources to heterogeneous resource scheduler, and heterogeneous resource scheduler dynamically adjusts and updates scheduling parameters in real time, based on comprehensive demand information and real-time status of resources in a refined multidimensional model.

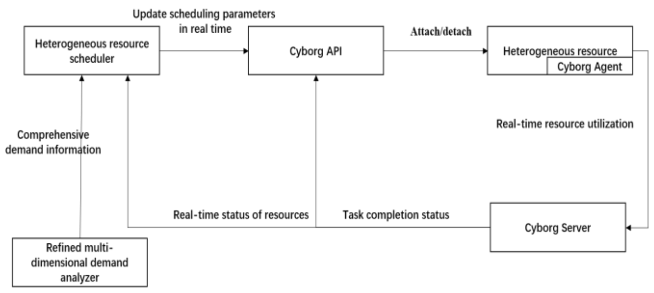
Update scheduling parameters in real time
Attach/detach
Heterogeneous resource scheduler
Cyborg API
Heterogeneous resource
Cyborg Agent
Comprehensive demand information
Real-time resource utilization
Real-time status of resources
Task completion status
Cyborg Server
Refined multi-dimensional demand analyzer

Fig. 6 Heterogeneous Resource Scheduling Model

IV. Conclusion

A. Summary

Edge Computing has attracted wide attention for its low latency and high bandwidth performance. With the advent of 5G era, various kinds of accelerated applications emerge at the historic moment. And the performance improvement brought by a single acceleration card is not enough to solve the challenges brought by the huge data scale computing task to the traditional architecture. So, the acceleration framework for Edge Cloud has become a research hotspot in the industry. At present, there are many related studies, and some results have been achieved in resource scheduling. However, the current Edge Computing frameworks are tightly coupled with specific application scenarios, causing the target resource pool highly customized, the whole service scenario dedicated to accelerating a certain computing task, and the utilization ratio of software and hardware resources low. The main contents of this paper are as follows: (1) aiming at the various and complex needs of edge-oriented cloud services, propose a refined multi-dimensional demand model to refine the demands in each service scenario, (2) propose an intelligent scheduling strategy for heterogeneous resources, which applies DRF (master resource scheduling algorithm) based on OpenStack-Cyborg acceleration framework. The scheduling strategy can realize the fair allocation of heterogeneous resources according to the priority of demand; (3) propose a heterogeneous resource scheduling model to realize the refinement of multi-dimensional demand model, collaboration between heterogeneous resource scheduling model and OpenStack-Cyborg interface. It can also provide direction for precise matching of service demands and heterogeneous resources.

Compared with previous research on edge-oriented computing framework, the advantages of this paper are shown in the following: (1) the intelligent framework for Edge Cloud has strong universality and high resource utilization rate, which can accurately match service demands and heterogeneous resources acceleration capabilities; (2) It can carry out refined multi-dimensional demands analysis for various business needs, and separate traditional needs from accelerated needs. And it can set weight parameters according to each demand to realize resource application according to priority. (3) Collaborative scheduling of heterogeneous resources through heterogeneous resource scheduling model and OpenStack-Cyborg acceleration framework can realize efficient and precise call of heterogeneous resources.

B. Expectation

This paper solves the problem of matching and scheduling between multiple demands and heterogeneous resources in the system architecture. On the basis of OpenStack-Cyborg framework, this paper explores the scheduling problem of heterogeneous resources, without studying the acceleration framework of each acceleration hardware more detail. In future studies, three aspects will be done: (1) For heterogeneous resource scheduling management, we will try to expand the acceleration framework of various kinds of accelerated hardware, such as Map Reduce and Tez framework under the accelerated hardware of FPGA, to optimize the resource scheduling strategy and ensure the fairness and effectiveness of resource allocation. (2) For terms of deployment strategy, this paper will study how to arrange network functions according to different service demands of users, including planning the deployment location of network functions and the forwarding path of network traffic. (3) For resource scheduling algorithm, we will continue to optimize the scheduling strategy, evaluate the utilization rate of DRF scheduling algorithm in heterogeneous resource pool, to optimize the scheduling algorithm, and achieve efficient utilization of resources. The computing power network is not simply a distribution of computing power information through the network. It also needs to be associated with computing power transactions, network ordering and other services, forming an architecture, to meet user demands and achieve the optimal allocation of various resources.

V. Acknowledgments

This work is supported by the National Key R&D Program of China (No. 2018YFB1800100).

References

[1] M. Alrowaily and Z. Lu, "Secure edge computing in iot systems: Review and case studies," in 2018 IEEE/ACM Symposium on Edge Computing (SEC), Oct 2018, pp. 440–444.

[2] Y. Xiong, Y. Sun, L. Xing, and Y. Huang, "Extend cloud to edge with kubeedge," in 2018 IEEE/ACM Symposium on Edge Computing (SEC), Oct 2018, pp. 373–377.

[3] H. Saxena and K. Salem, "Edgex: Edge replication for web applications," in 2015 IEEE 8th International Conference on Cloud Computing, June 2015, pp. 1041–1044.

[4] Z. Rao, "Research on mobile edge computing and video analysis application," Master's thesis, Beijing University of Posts and Telecommunications, 2017.

[5] M. Liu, "The research of resource collaborative optimization between terminal and base-station in mobile edge," Master's thesis, Beijing University of Posts and Telecommunications, 2018.

[6] W. Wu, "Research on fpga heterogeneous cluster system based on yarn," Master's thesis, Huazhong University of Science and Technology, 2016.