Research on Key Technologies of End-Side Computing Force HCI Network Based on Multi-Granularity and Multi-Level CAD End-Side Computing Force Scheduling

Hengjiang Wang[1], Fang Cui[2], Mao Ni[3]* and Ting Zhou[4]

[1,2,3,4]China Mobile Group Device Co., Ltd., Beijing, 100053, China

[1]wanghengjiang@cmdc.chinamobile.com,[2]cuifang@cmdc.chinamobile.com,[3]nimaoch@163.com
[4]zhouting@cmdc.chinamobile.com

Corresponding author: Mao Ni, nimaoch@163.com

Abstract. The traditional independent and single "cloud rendering" platform is far from meeting the market demand, and the massive social computing power has not been effectively utilized. As the demand for efficient and responsive design processes increases, leveraging end-side computing can significantly enhance Human-Computer Interaction (HCI) experiences by optimizing resource allocation and reducing latency. Based on multi-granularity and multi-level end-side computing power scheduling algorithms, this paper studies the end-side computing power network technology. Aiming at the complex and changeable application types and different personalized needs of the mobile Internet of Things, it proposes an efficient management method based on the graph theory model and deep learning method to slice the core and edge parts of the Internet of Things, which can meet the specific needs of different applications and improve the quality of service and user experience under the condition of limited resources. Through experimental research, the model proposed in this paper has specific effects. The findings indicate that a well-designed HCI network, supported by advanced CAD technologies, significantly improves end-side computing systems' efficiency and user experience.

Keywords: multi-granularity and multi-level CAD; end-side computing force; Human-Computer Interaction; Computer-Aided Design.
DOI: https://doi.org/10.14733/cadaps.2025.S6.131-146

1 INTRODUCTION

In cloud edge aggregation computing, the increasing computing demand promotes the continuous expansion of the cloud data center scale. Effective load forecasting is crucial in realizing flexible resource allocation and can provide decision-making references for high-quality expansion schemes of cloud data centers. Meanwhile, the load change of a cloud data center can reflect the running state of the current data center server cluster, and the resource utilization rate of a cloud data center in the current and future period is analyzed according to the actual running situation of its components [2].

By combining high-precision load forecasting models, cloud service providers can dynamically adjust cloud data centers' online server size, related hardware settings, resource configuration, etc., improving resource utilization, reducing energy consumption, and saving costs. As the primary user of cloud data center resources, the load of cloud data centers is driven by user behavior, including but not limited to computing-intensive, IO-intensive, and other types of computing tasks, which results in significant fluctuations in their load intensity. At the same time, the demand for related computing resources is also dynamically changing [12]. Due to the development of emerging technologies and differences in the structure and types of services provided by different cloud computing centers, load forecasting for cloud-based multiple data centers has become increasingly complex. Currently, most of the existing load forecasting methods focus on a single data center, using a single model method for prediction [9] or integrating the prediction results of various models based on specific rules [11] to select the optimal prediction model for different scenarios. The types of computing tasks running on server clusters are constantly changing, system resource usage is also fluctuating, and data centers have distinct differences. The mathematical methods for building load prediction models often need to be adjusted promptly to avoid loss of accuracy, resulting in these algorithms not being suitable for the current cloud-based multi-data center environment. This requires designing a universal method with strong applicability and high robustness for load forecasting in cloud data centers.

Due to the inability to predict user computing needs, the system selects the optimal target server to process computing task requests in a dynamically changing resource environment. Suppose there is no reasonable task offloading mechanism. In that case, it will cause frequent network congestion, increase the number of failed tasks, extend task execution time, and decrease QoS quality. Currently, existing task-offloading methods often use optimization or multi-constrained joint optimization methods [15]. These algorithms utilize scheduling offloading decisions to minimize task offloading energy consumption or latency but ignore the collaborative work effect of multiple resources in edge environments. Traditional offline optimization methods only optimize system performance from a single perspective of latency or energy consumption and cannot adapt to the highly dynamic changes of edge environment resources. A practical task offloading strategy directly affects the processing ability of edge systems for user tasks, which helps to improve system performance and has significant research significance. The task offloading strategy should comprehensively consider various factors, such as the performance of the mobile device itself, the usage of edge servers and remote cloud servers, and the current network link quality to determine whether the computing tasks of the current terminal device need to be offloaded and to which specific server [16]

The essence of load forecasting is to analyze historical data within a certain period in the past and calculate the load size at a particular moment or period in the future. Traditional time series prediction includes continuous and discrete predictions; load prediction belongs to continuous prediction, namely numerical prediction [6]. The order of numerical values can affect the results of load prediction. If the magnitude of the values remains unchanged, but the order of values changes, the predicted results may be completely different.

Mathematical statistical model: A mathematical statistical model represented by the ARIMA model [20]. When it is necessary to predict the values of a load sequence node, the model usually needs to obtain the relevant values of each node in the period before that time. Because load sequences are usually non-stationary, analyzing and observing the trend of load prediction can obtain their long-term, seasonal, or periodic trends [18]. Due to the correlation between load size and the status of various components in the data center, the load prediction trend is challenging to observe directly. Through sequence decomposition methods commonly used in signal processing, such as the

Hale wavelet transform [17], the load sequence can be decomposed to observe the changing trend in different frequency bands, thus obtaining the correlation between input values and the final load. By manually adjusting parameters, the mathematical parameters required for the model can be set to analyze the future trend of load changes. In addition to the ARIMA model, mathematical and statistical models that can be used for load series prediction include naive prediction, exponential smoothing prediction, Holt-Winters prediction method, and weighted moving average method.

Machine learning model: A machine learning model represented by SVR [13]. Support vector regression is a vital application branch of support vector machine models. In two-dimensional space, SVR fits data with acceptable errors defined in the model by finding a suitable curve. In multidimensional space, SVR regression constructs a regression plane to achieve the closest distance from all data in a set to that plane. These methods are usually data-driven. They use historical data to train the model, collect time characteristics and sample values, learn the relationship between attributes and values through supervised learning, and obtain the random dependency relationship between past and future loads.

Deep learning model: As represented by reference [14], an RNN model is a neural network used to process data sequences. Traditional convolutional neural networks are considered unsuitable for time series prediction due to limitations in the size of convolutional kernels, which prevent them from obtaining the connection between the values before and after the sequence data.

In studying load prediction algorithms using statistical learning methods, reference [8] used the ARIMA model to predict the load size in cloud computing environments. It analyzed the impact of workload changes on cloud application QoS. Reference [1] used a Kalman filter based on the maximum entropy criterion to predict and analyze the CPU state in virtual machines, thereby dynamically adjusting resource allocation. Reference [3] applies the Bayes method to load prediction to predict the average load over long time intervals. These methods carry out load forecasting based on preset formulas, which are simple and highly interpretable. However, their forecasting effect is limited by the setting of model parameters, and they need rich experience in parameter adjustment to artificially select specific parameter values. It reduces the computational workload, fails to consider the changing characteristics of load prediction modes, and does not conduct an in-depth analysis of the influencing factors of load changes, neglecting the possible connections between relevant resources within the system, resulting in low prediction accuracy. With the continuous development of neural network technology, more and more neural network methods are being applied to load forecasting research. Reference [5] introduced Artificial Neural Networks (ANN) technology to load forecasting problems, but the effectiveness of its prediction results significantly decreased in large cloud data centers. Reference [7] applied the Radial Basis Function (RBF) neural network to the energy consumption prediction problem in data centers, proving that the improved RBF model has better fitting and adaptability compared to the BP network in short-term energy consumption prediction problems. Based on the back-propagation neural network, literature [10] applied the Long Short Term Memory (LSTM) neural network to the load forecasting problem to mine the hidden relationship between the front and back of time series data. Although these neural network methods do not have advantages in the interpretability and response speed of the model, their models have more substantial expressive power and higher accuracy in handling complex load changes. However, it is still unsuitable for predicting complex states with frequent load fluctuations and interference from external uncertain factors. In the model fusion method, reference [4] proposed a data combination processing method GMDH and an integrated wavelet decomposer to reduce nonlinear errors at different time-frequency scales and predict the actual load size for each continuous future time interval. Reference [19] analyzed the effectiveness of several different load prediction models, including support vector machines, Holt-Winters, and genetic algorithms. It proposed an automatic decision model to select the optimal algorithm in various scenarios.

Combined with a multi-granularity and multi-level end-side computing force scheduling algorithm, this paper studies end-side computing force network technology proposes the algorithm model and applies it to improve end-side computing force network lifting.

2 SLICING MODEL OF MOBILE INTERNET OF THINGS BASED ON MULTI-LAYER GRAPH THEORY

2.1 Model Constraints and Basic Definitions

Multi-layer graph theory is a complex scene in the research field of graph theory, which is defined as a node or an edge not only has one attribute but has multiple attributes and the edge and node with the same attribute are divided into the same layer, and various layers will be divided due to the increase of attributes. The analysis and research on multiple layers in a graph are defined as multi-layer graph theory. In the slice environment of the mobile Internet of Things, a node may deploy functional components of various slices, and a link may also belong to multiple slices. Therefore, a layer set $L = \ L_1, L_2, ..., L_d$ Is defined, A multi-layer graph G with vertex set V, edge set E, and layer set L is defined as $G = (\{V_i\}_{i=1}^{|L|}, \{E_i\}_{i=1}^{|L|}, L)$, where V_i and E_i E Represents corresponding vertex sets and edge sets belonging to different layers. Relevant parameters are defined as follows: $R_{total} = \ C_{total}, D_{total}, T_{total}$, where R is the overall available resources of the Internet of Things system, C represents computing resources, D represents storage resources, and T represents network transmission resources. $C_{total} = \{c_1, c_2, c_3, ..., c_n \mid c_i \in C\}$ represents the available computing resources of the system, where c_i is the available computing resources of a cloud computing node where virtual functions can be deployed. $D_{total} = \{d_1, d_2, d_3, ..., d_n \mid d_i \in D\}$ represents the available storage resources of the system, where d_i is the available storage resources of a cloud computing node. $T_{total} = \{t_1,\ \ t_2,\ \ t_3,\ \ ...,\ \ t_n \mid t_i \in T\}$ represents the available network transport resources of the system, where t_i Is the available network transport resources between different cloud computing nodes connected? The transmission delay A Particular link can be calculated according to the amount of data transmitted and the path's network bandwidth. The propagation delay $_{prop}$ is determined by the transmission time of the optical signal in the link, which can be calculated by dividing the link length by the speed of light. However, the switch's processing capacity determines the queuing and processing delays, so this paper will not consider them for now. In this paper, only transmission delay and propagation delay are considered. Therefore, there is a delay. $l_i = l_{trans} + l_{prop}$, That is, the total delay is the sum of the transmission delay and the propagation delay of the link. In the model, considering the bandwidth of transmission resources and the transmission path length $t_i = \ B_i, \mathrm{P}_i\ , \mathrm{B}_i$ represents link bandwidth and P_i represents link length. The system generates a new slice. s_i according to the requirements. The slice is defined as $s_i = \ A_i, \mathrm{R_i}, \mathrm{G}_i\ :\ A_i$ is the specific application, G_i is the applied resource, and G_i is the layer corresponding to the slice generated for the application after applying for the resource. G_i represents the total resources acquired by slices s_i hosting an application A_i, described by a graph G_i .

All slices of the Internet of Things system are defined as $S = \{s_1, s_2, s_3, ..., s_n \mid s_i \in S\}$, where s_i Is a specific slice. According to the above slice definition, tuition with limited overall resources means $\sum_{i=1}^{n} R_i \leq R_{total}$, and R_{total} is available resources for the whole system.

The graph G_i will be automatically generated according to the business's specific needs under the restricted resources G_i Of the slice. Therefore, the attributes of node v and edge e in the graph G_i are defined as $v = \ C_v, D_v \ , e = \ B_e, P_e$, where C_v and D_v represent the computational and storage resources allocated for node v in the slice s_i, respectively, and Be and Pe represent the available bandwidth and link length of link e, respectively.

All resources: Available resources s Will limit the G_i of a slice; that is to say, all nodes' computing and storage resources do not exceed the allocable computing and storage resources. The sum of bandwidths of all links in the slice does not exceed the total bandwidth of the slice; that is, there are:

$$\sum_{n=1}^{m} v_n(c,d) \leq R_i(\mathrm{C},\mathrm{D}) \tag{1}$$

$$\sum_{n=1}^{m} e_n(b) \leq R_i(\mathrm{B}) \tag{2}$$

Whether during slice initialization or slice running, when allocating resources for slices, there is always:

$$\underset{n}{argmax} \sum_{i=1}^{n \sum_{itotal}} min \tag{3}$$

Formula 3 represents that when allocating resources for all slices, as few resources as possible are allocated for each slice on the premise of meeting application requirements.

If we assume that the amount of data to be transmitted and computed by an application is w, the sum of computational resources of each virtual function in the slice is C, and the total link resource is B, there is a delay $L = a\frac{w}{C} + \beta\frac{w}{B}$. Because the computational and transmission delays are not in the same order of magnitude, constants α and β are used to adjust. Therefore, increasing the computing and link resources is necessary to reduce the delay while the amount of transmitted data is constant. When all the resources of a particular slice are limited, the objective function of delay minimization is:

$$\underset{i}{min}\, L = \frac{\alpha}{max\sum_{n=1}^{m} c_n} + \frac{\beta}{max\sum_{n=1}^{m} b_n} \mathrm{s.t.} \sum_{n=1}^{m} c_n \leq R_i \ C \ \sum_{n=1}^{m} b_n \leq R_i \ B \tag{4}$$

2.2 Model and Algorithm Design

To meet the complex and diverse business requirements in 5G and 5G Beyond the Internet of Things, different network functions may be deployed at the core position or run at the edge position, and all other functions will be connected through a suitable secure link to provide corresponding services for the services carried by different slices in the form of virtual function chain. This chapter uses

GraphTheory to describe an abstract mobile Internet of Things slices. It analyses the deployment, operation, and optimization of slices using multi-layer weighted graphs G_i V,E .

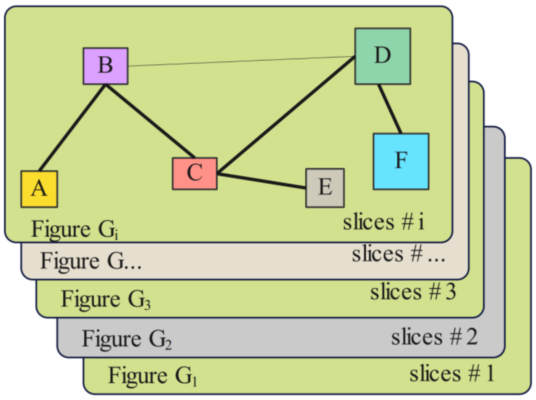


Figure 1: Network slicing based on a multi-layer weighted graph.

In Figure 1, the graph G_i is an abstract model of slice I, and different nodes and paths have corresponding characteristic attributes. In the slice represented by each layer, the performance of all nodes and links can be reflected. For example, the related performance of a node v_i is expressed by its radius. The larger the node (radius), the more resources and the stronger the performance of the physical or virtual platform running the virtual function. On the contrary, the smaller the node, the weaker its processing capacity. The weight w_{ij} of each edge $e_{ij} = v_i$,v$_j$ is expressed by the thickness of its line segment. The thicker the line segment of an edge, the more network resources, the larger the bandwidth, and the lower the delay used to transmit data between nodes. On the contrary, the thinner the edge, the smaller the bandwidth and the higher the delay.

If we assume that the current node is v_i , The end node is V_j,WV_j is the resource state weight of the end node and the path of connection v_i and V_j is R_{ij} , and WR_{ij} is the resource state weights of the current path, there are:

$$W_{ij} = WRW_{ij} * V_j \tag{5}$$

The wandering probability of each path is:

$$P_{ij} = \frac{W_{ij}}{\sum_{j=1}^{\deg(V_i)} W_{ij}} \tag{6}$$

According to the random walk nature of the graph, there are:

$$\sum_{j=1}^{\deg(V_i)} P_{ij} = 1 \tag{7}$$

The mathematical description formula for generating a countermeasure network is as follows:

$$\min_{G}\max_{D} V(D,G) = E_{x \circledR P_{data}(x)}\left[\log D(x)\right] + E_{z\sim P_z(Z)}\left[1-\log D(G(z))\right] \tag{8}$$

From the point of view of discriminator D, D wants to distinguish actual samples from false samples as much as possible, so it wants $D(x)$ to be as large and $D(G(z))$ to be as small as possible, that is, V (D, G) is as large as possible. From the generator's point of view, G wants to fool D as much as possible; that is, it wants $D(G(z))$ to be as big as possible, that is, V(D, G) to be as small as possible. The two models oppose each other and finally reach the global optimum. In resource prediction, through continuous iteration, generator D continuously generates more realistic data to predict the resource demand of slices.

The specific mathematical description is as follows:

$$Q_{t+1}(S',A') = Q_t^p(S,A) \tag{9}$$

The transition between the state S at time t+1 and the new joint action set A' occurs after joint action A executes the resource allocation strategy π in the state S at time t. Chapter 5 details the specific algorithm and implementation process.

To meet the demand for low delay in the mobile process of various terminals, the overall delay of the system is divided into two parts: transmission delay and data processing delay, which are expressed by the following formula:

$$L = \alpha L_{px} + \beta L_{tx} \tag{10}$$

Because the processing and transmission delays are not of the same order of magnitude, the parameters alpha and beta are used for adjustment.

If we assume that the total amount of data to be processed and transmitted in task T of the current slice S is D_t^T , The current delay can be expressed as:

$$L_t = \alpha \frac{D_t^T}{R_{px}} + \beta \frac{D_t^T}{R_{tx}} \tag{11}$$

The formula represents the processing and transmission resources used to process and transmit data. Since all the data to be processed are equal and the goal is to obtain the minimum delay, the above formula can be changed as follows:

$$\min L_t = \frac{\alpha}{\max R_{px}} + \frac{\beta}{\max R_{tx}} \tag{12}$$

In this paper, the graph neural network algorithm is used to infer the available resources of each node. The mathematical expression is as follows:

$$f_v^k = \sigma\left(\left[W_k \cdot CONCAT\left(LSTM\left(F_u^{k-1}, \forall u \in N(v)\right), f_v^{k-1}\right)\right]\right) \tag{13}$$

In the above form and subscript base, cap F of a node at the current moment and F_u^{k-1} is the state of all neighboring nodes at the previous moment. The LSTM algorithm obtains the state of itself and its adjacent nodes at the last moment through multi-layer iteration. The specific algorithm and implementation process are detailed in Chapter 6.

2.3 Slicing Resource Management of Mobile Internet of Things Based on Deep Learning

Figure 2 depicts the slice resource management architecture of the mobile Internet of Things based on deep learning, which is mainly composed of four layers: slice layer, control layer, management layer, and learning layer. The slice layer has two functional architecture modules: the data exchange and virtual function modules. The main task of the switch is to forward data traffic according to flow table entries. The SDN controller in the control layer controls the specific situation of the whole network through the OpenFlow protocol. In the virtual function component module, the virtual function modules of mobile Internet core, such as MMA, vSGW, vPGW, and vHSS, are run on the virtualization platform based on the LXC container.

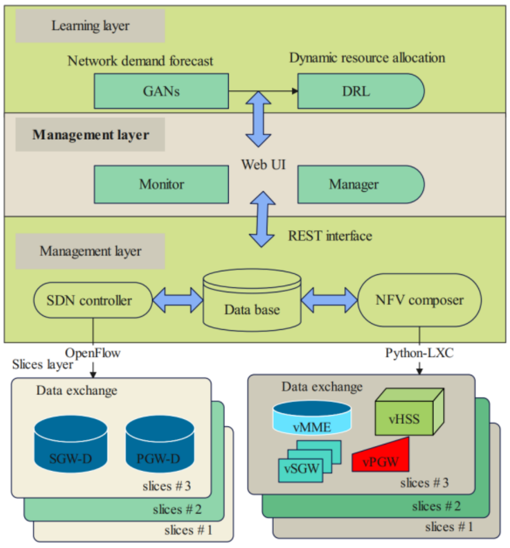


Figure 2: Slicing mobile Internet of Things architecture based on deep learning.

We first introduce the application of generating a countermeasure network model in network resource demand prediction.

1. Problem description: In the high-speed mobile Internet of Things environment, administrators must accurately predict resource demand to ensure the quality of the end-user experience.

DeepSlicing uses network downlink rate, uplink rate, and network delay metrics for network load forecasting. DLt 、 Ult, and Dt represent the corresponding t parameters at a particular time: empty, empty, $X_t = \{DL_t, UL_t, D_t\}^T$ represents corresponding network parameter metrics at time t. We use Y_t to redescribe a work metric case in which all slices in a time series t(t=1,2,..., T) have a time slice of one minute, where T is the maximum time metric. Here, random basic data sets X X= $X_1, X_2, \ldots, X_T$ and slice network measurement situation history data sets Y Y= $Y_1, Y_2, \ldots, Y_T$ are used. The goal of the model is to predict the network slice measurement for the next time slice Y_{T+1}. Because 24 hours is 1440 minutes, this article sets the maximum time T to 1440.

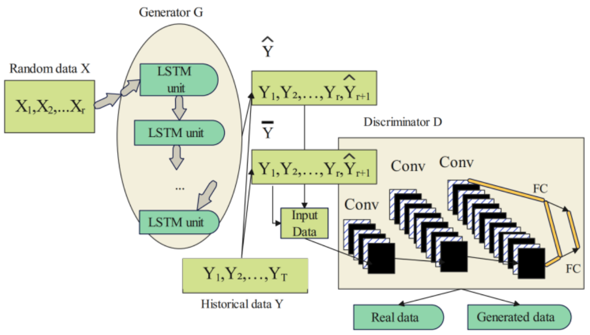


Figure 3: Resource demand prediction model based on generating countermeasure network.

2. Prediction model: Figure 3 shows the prediction module for generating countermeasure network demand in the DeepSlicing model proposed in this paper. Because network traffic is based on time series, the LongShortTermMemory (LSTM) network is used as the prediction model in generator G. The traffic data cap Y that subdues the traffic data $\hat{Y}_{T+1}$ of the next time slice is predicted and generated through input data X.

$$\hat{Y}_{T+1} = G \;\; X \tag{14}$$

LSTM is a deep learning model that can rely on learning for different time lengths. Moreover, LSTM performs well in many changeable problems, such as natural language text extraction, handwriting recognition, and power load forecasting.

The discriminator D is realised by a convolution neural network model, which performs the convolution operation of multidimensional data (such as DLt, ULt, Dt). The input data of discriminator D in D, the deep slicing prediction module, is composed of accurate historical data $\bar{Y} = Y_1, Y_2, \ldots, YT, Y_{T+1}$ or data ${}^nG \;\; \hat{Y} = Y_1, Y_2, \ldots, YT, Y_{T+1}$ generated by the generator.

A convolution neural network is a feed-forward network with excellent image and video recognition, recommendation systems, and natural language processing performance.

3. Model training: The generator and discriminator (G, D) are trained in two steps using the StochasticGradientDescent (SGD) algorithm.

Training of generator G: ((X, Y) is sample data.) To "confuse" discriminator D as much as possible, generator G needs to continuously reduce the confrontation loss so that discriminator D cannot judge whether the data is accurate or generated. The system puts real data ${}^nY$ classified into 1, classifies the generated data Y into 0, and d defines the countermeasure loss function of generator G as L^G_{adv}

.

$$L^G_{adv} \;\; \hat{Y} \;\; = L_{sce} \;\; D \;\; \hat{Y} \;\; ,1 \tag{15}$$

Among them, L_{sce} is the SigmoidCross-Entropy loss function, which is defined as:

$$L_{se}\ A,B\ = -\sum_i B_i\ sigmoid\ A_i\ + 1 - B_i\ \log(1 - sigmoid\ A_i\ ) \tag{16}$$

Only using the minimum confrontation loss function in the natural environment cannot satisfy prediction accuracy. Generator G may use the wrong samples to confuse discriminator D, and discriminator D will use the wrong countermeasure samples to learn continuously, which will cause the model's failure. Therefore, generator G needs to reduce error loss and use Lp loss:

$$L_p\left(\overset{n}{Y},\widehat{Y}\right) = \overset{n}{Y} \text{-} \hat{Y}_p \tag{17}$$

Among them, p=1 or p=2.

We must define a directional prediction loss function because mobile traffic prediction is closely related to resource allocation and user experience quality. L_{dpl} :

$$L_{dpl}\left(\overset{n}{Y},\widehat{Y}\right) = \left|sign\ \widehat{Y}_{T+1} \text{-} Y_T\ \text{-} sign\ Y_{T+1} \text{-} Y_T\ \right| \tag{18}$$

Among them, the sign is a symbolic function.

Then, we combine the above three related loss functions of generator G with appropriate parameters α_{adv}, α_p , and α_{dpl}, and The final loss function is:

$$L_G\ X,Y\ = a_{adv} L_{adv}^G\ \widehat{Y}\ + a_p L_p\left(\overset{n}{Y},\widehat{Y}\right) + a_{dpl} L_{dpl}\left(\overset{n}{Y},\widehat{Y}\right) \tag{19}$$

Training of discriminator D: ((X, Y) are different data samples). Because the role of D is to judge whether the input 3D data is real data $\overset{n}{Y}$ or generated data $\hat{Y}$, Its target loss function is the same as that of the generator. Therefore, while keeping the weight of generator G unchanged, we use one-step SGD to minimize the target loss function. X,Y :

$$L_D\ X,Y\ = L_{adv}^G\left(\overset{n}{Y},\widehat{Y}\right) = L_{sece}\ D\ \widehat{Y}\ ,0\ + L_{sece}\left(D\left(\overset{n}{Y}\right),1\right) \tag{20}$$

4. Training data set: To simulate the real Internet of Things data, this paper's training data is downloaded from 4G data collected by the Department of Computer Science of Cork University in Ireland from two significant operators in Ireland.

Figure 4 depicts the multi-agent, multi-level reward deep reinforcement learning model proposed by DeepSlicing for slice resource allocation of mobile Internet of Things. DeepSlicing's resource configuration module is implemented by DeepQ-learningNetwork (DQN). As shown in Figure 6, it is defined that. S_t is the corresponding observation state of the algorithm until time t . At is the joint action set executed in the state? S_t and R_t is is the multi-level reward set obtained by the agent after executing the joint action A_t in the state S_t .

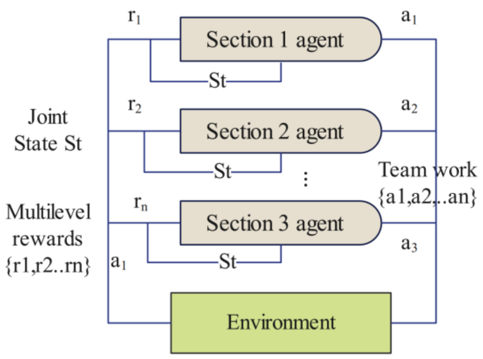


Figure 4: Deep reinforcement learning algorithm of multi-agent and multi-level reward.

Meanwhile, we take DNN as the action state value function. $Q\ \ S_t, \mathrm{A}_t$. Optimal learning of the action state value function Q is realized according to the iteration in formula (21):

$$\begin{cases} Q_{k+1}\ \ S_t, A_t\ \ = Q_k\ \ S_t, A_t\ \ + \alpha_k E_k \\ E_k = R_t + \gamma \underset{A \in \Lambda}{max} Q_k\ \ S', A'\ \ - Q_k\ \ S_t, A_t \end{cases} \tag{21}$$

From the above formula, the method to realize the approximation of the action state value function is as follows:

$$Q_{k+1}\ \ S_t, A_t\ \ \approx Q_k\ \ S_t, A_t \tag{22}$$

Then, E_k Needs to tend to 0, that is:

$$R_t + \gamma \underset{A' \in \Lambda}{max} Q_k\ \ S', A'\ \ - Q_k\ \ S_t, A_t\ \ \text{--}0。 \tag{23}$$

For each iteration k, the parameter update can be achieved by minimizing the objective function shown in formula (23).

$$minE = |\min(R_t + \gamma \underset{A' \in \Lambda}{max} Q_k\ \ S', A'\ \ - Q_k\ \ S_t, A_t\ \ )| \tag{24}$$

Therefore, this paper takes minE in the above equation as the error function, uses the SGD algorithm to update the parameters in DNN, and obtains the optimal solution of the action state value function.

3 END-SIDE COMPUTING FORCE NETWORK MODEL BASED ON MULTI-GRANULARITY AND MULTI-LEVEL END-SIDE COMPUTING FORCE SCHEDULING

Suppose we want to determine the relationship between the computing power level and the macro digital economy's development. In that case, we need to build a systematic framework of computing power index from three dimensions: computing power environment, computing power scale, and computing power application, as shown in Figure 5, to make a comprehensive evaluation.

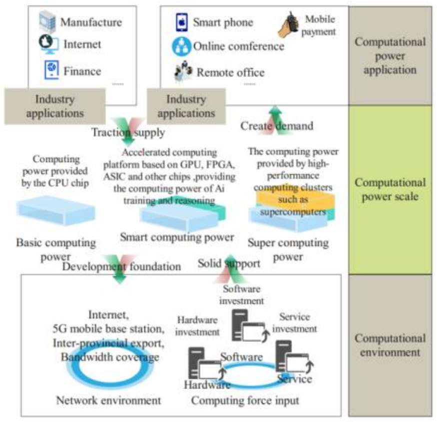

Manufacture
Internet
Finance
Smart phone
Mobile payment
Online comference
Remote office
Computational power application
Industry applications
Industry applications
Traction supply
Create demand
Computing power provided by the CPU chip
Accelerated computing platform based on GPU, FPGA, ASIC and other chips ,providing the computing power of Ai training and reasoning
The computing power provided by high-performance computing clusters such as supercomputers
Computational power scale
Basic computing power
Smart computing power
Super computing power
Development foundation
Solid support
Software investment
Internet, 5G mobile base station, Inter-provincial export, Bandwidth coverage
Hardware investment
Service investment
Software
Hardware
Service
Network environment
Computing force input
Computational environment

Figure 5: Computational power index framework.

The functional architecture of the computing power network is divided into four modules: computing power network resource layer, computing power network control layer, computing power network service layer, and computing power network arrangement management layer, as shown in Figure 6. The service provider layer mainly realizes user-oriented services, and its atomic functional capabilities are open. The network control layer mainly realizes the routing of computing and network multidimensional resource integration through the network control plane. The computing power network resource layer primarily provides computing power resources, storage resources, and network forwarding resources. It combines them with the computing processing ability and network forwarding ability in the network to realize the transmission and flow of various computing and storage resources. The computing power management orchestration layer mainly solves the problems of registration, modeling, management, orchestration, and security of heterogeneous computing power resources and service/function resources.

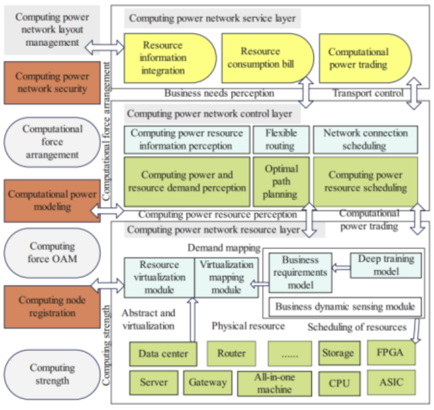

Computing power network layout management
Computing power network security
Computational force arrangement
Computational power modeling
Computing force OAM
Computing node registration
Computing strength
Computational force arrangement
Computing strength
Computing power network service layer
Resource information integration
Resource consumption bill
Computational power trading
Business needs perception
Transport control
Computing power network control layer
Computing power resource information perception
Flexible routing
Network connection scheduling
Computing power and resource demand perception
Optimal path planning
Computing power resource scheduling
Computing power resource perception
Computational power trading
Computing power network resource layer
Demand mapping
Resource virtualization module
Virtualization mapping module
Business requirements model
Deep training model
Business dynamic sensing module
Abstract and virtualization
Physical resource
Scheduling of resources
Data center
Router
......
Storage
FPGA
Server
Gateway
All-in-one machine
CPU
ASIC

Figure 6: Overall functional architecture diagram of the computing power network.

The centralized scheme architecture of multi-granularity and multi-level computing power networks comprises four parts (Figure 7). (1) Computing power network management arrangement system. The resource management and scheduling system of the computing power network flexibly schedules computing power resources according to business requirements, which can meet real-time business requirements and improve the utilization rate of computing power. (2) Empowering platform. It empowers user business deployment by providing an enabling platform for AI businesses. (3) Edge/core DC. The business deployment node includes computing power resource infrastructure and NFV infrastructure. (4) Network infrastructure. The network infrastructure connecting users, edge cloud, and core cloud consists of an SDN controller on the control plane, traditional network management, and network equipment on the forwarding plane.

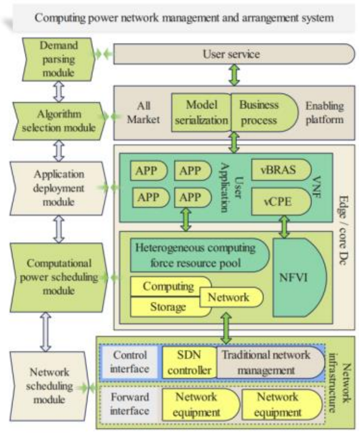
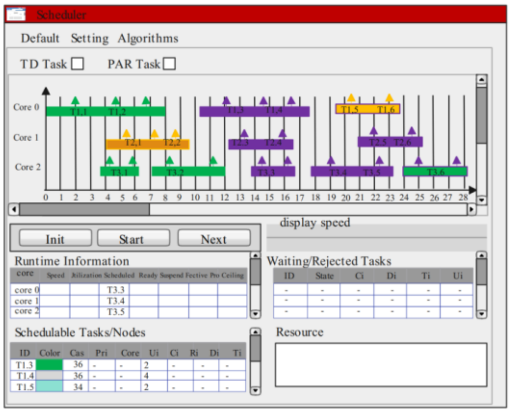


Figure 7: The centralized scheme architecture of multi-granularity and multi-level computing power network.

There are two main aspects to testing the task scheduling simulation module: one is to test the new function points, and the other is to test the parallel task scheduling simulation process. The two tests can use the same set of functions, except that one is added manually, and the other is embedded in a slot function, which can automatically complete a series of configurations as long as it is triggered. The attributes of tasks appear in bold italics, and the remaining areas represent related attribute information, such as nodes contained by functions in the task collection. At present, the priority of a node is set to be the same as the first level of its task, and its priority can be modified according to different scheduling strategies in the later stage. The task set is verified in a 3-core and 4-core environment. The simulation results are shown in Figure 8 and Figure 9.



Runtime Information

core	Speed	Utilization	Scheduled	Ready	Suspend	Fective Pro	Ceiling
core 0			T3.3				
core 1			T3.4				
core 2			T3.5				

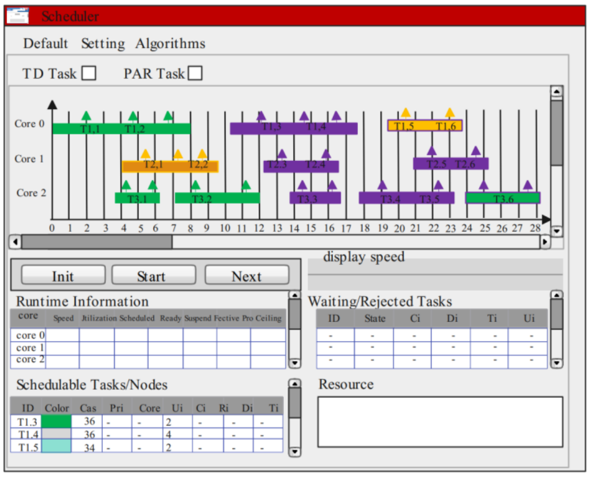
Waiting/Rejected Tasks

ID	State	Ci	Di	Ti	Ui
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-

Schedulable Tasks/Nodes

ID	Color	Cas	Pri	Core	Ui	Ci	Ri	Di	Ti
T1.3	🟩	36	-	-	2	-	-	-	
T1.4	⬜	36	-	-	4	-	-	-	
T1.5	🟦	34	-	-	2	-	-	-	

Figure 8: Schematic diagram of 3 core operations.



Runtime Information

core	Speed	Jtilization	Scheduled	Ready	Suspend	Fective	Pro Ceiling
core 0							
core 1							
core 2							

Waiting/Rejected Tasks

ID	State	Ci	Di	Ti	Ui
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-

Schedulable Tasks/Nodes

ID	Color	Cas	Pri	Core	Ui	Ci	Ri	Di	Ti
T1.3	🟩	36	-	-	2	-	-	-	
T1.4	⬜	36	-	-	4	-	-	-	
T1.5	🟦	34	-	-	2	-	-	-	

Figure 9: Schematic diagram of 4 core operations.

The key technologies of end-side computing force network based on multi-granularity and multi-level end-side computing force scheduling proposed in this paper are verified, and the algorithm model proposed in this paper is explored to evaluate the improvement effect of end-side computing force. Finally, the evaluation results shown in Table 1 are obtained.

NO.	*Computational power level*	*NO.*	*Computational power level*	*NO.*	*Computational power level*
1	*83.0767*	*10*	*88.7048*	*19*	*84.4737*
2	*87.3315*	*11*	*83.4999*	*20*	*84.4954*
3	*87.8191*	*12*	*82.2114*	*21*	*86.6563*
4	*90.6750*	*13*	*91.9596*	*22*	*89.5421*
5	*85.0401*	*14*	*86.9620*	*23*	*90.9322*
6	*88.3464*	*15*	*83.9031*	*24*	*87.8004*
7	*90.1623*	*16*	*88.7748*	*25*	*88.7463*
8	*86.8693*	*17*	*87.2608*	*26*	*91.7801*
9	*85.8927*	*18*	*83.5348*	*27*	*87.8081*

Table 1: Evaluation of improvement effect of key end-side computing power network technologies based on multi-granularity and multi-level end-side computing power scheduling.

From the above experimental research, the critical technology of end-side computing force network based on multi-granularity and multi-level end-side computing force scheduling can effectively improve the level of end-side computing force and promote the performance of end-side equipment.

4 CONCLUSIONS

With the rapid development of 5G, high speed and low latency are the main technical characteristics of the network. The increasing weight of wireless access promotes the growth of mobile edge computing, which enables the generation, processing, and application of services to be completed locally instead of relying solely on remote centralized units. In future network applications, the impact of the access side will become more and more profound, the requirements of service application speed and latency will become higher and higher, and the role of mobile edge computing will become more prominent. Moreover, the computing power network architecture will take the deep integration of network and computing as the engine. This paper focuses on scalable and secure deployment of mobile Internet of Things slices according to the specific requirements of different services, predictable and efficient utilization of Internet of Things slice resources, further improvement of user experience, and optimization of high-performance edge slices during terminal mobility. In addition, an end-side computational force network model based on multi-granularity and multi-level end-to-end computational force scheduling is proposed. The experimental results show that the model proposed in this paper has specific effects. End-side computing force networks in the automobile sector require extensive research to optimize vehicle systems. This study intends to increase performance and reliability by emphasizing multi-granularity computing, multi-level scheduling, and efficient resource management. Integrating these technologies using optimization algorithms, communication protocols, and security measures promises to improve functionality and user safety and is ultimately successful.

Hengjiang Wang, https://orcid.org/0009-0002-0900-027X
Fang Cui, https://orcid.org/0009-0008-7446-7195
Mao Ni, https://orcid.org/0000-0002-8951-0749
Ting Zhou, https://orcid.org/0009-0002-3779-0299

REFERENCE

[1] Chang, Z.; Liu, S.; Xiong, X.; Cai, Z.; Tu, G.: A survey of recent advances in edge-computing-powered artificial intelligence of things, IEEE Internet of Things Journal, 8(18), 2021, 13849-13875. https://doi.org/10.1109/JIOT.2021.3088875
[2] Chinweoke, O. U.; Christopher, I. W.: Load Evaluation with Fast Decoupled-Newton Raphson Algorithms: Evidence from Port Harcourt Electricity, Advances in Science, Technology, and Engineering Systems Journal, 5(5), 2020, 1099-1110. https://doi.org/10.25046/aj0505134
[3] Dong, Y.; Zhang, F.; Li, X.; Zhang, L.; Yu, J.; Mao, Y.; Jiang, G.: Nonlinear Load Harmonic Prediction Method Based on Power Distribution Internet of Things, Scientific Programming, 2021, 2021, 1-12. https://doi.org/10.1155/2021/9978900
[4] Elfatih, N. M.; Hasan, M. K.; Kamal, Z.; Gupta, D.; Saeed, R. A.; Ali, E. S.; Hossain, M. S.: Internet of vehicle's resource management in 5G networks using AI technologies: Current status and trends, IET Communications, 16(5), 2022, 400-420. https://doi.org/10.1049/cmu2.12315
[5] Irfan, M.; Ali, F.; Muhammad, F.; Rahman, S.; Armghan, A.; Khan, Y.; Gommosani, M. E.: Impairments Approximations in Assembled mmWave and Radio Over Fiber Network, CMC-Computers Materials & Continua, 73(3), 2022, 4885-4895. https://doi.org/10.1155/2021/9978900
[6] Lv, Z.; Qiao, L.; Verma, S.: AI-enabled IoT-edge data analytics for connected living. ACM Transactions on Internet Technology, 21(4), 2021, 1-20. https://doi.org/10.1145/3421510
[7] Mohamed, S. A.; Luo, N.; Pujol, T.; Pacheco, L.: Voltage Sourced Converter (VSC) based on multiple FACTS controllers for improving power quality: Renew, Energy Power Qual. J, 1(16), 2018, 65-70. https://doi.org/10.24084/repqj16.213
[8] Moons, B.; Karaagac, A.; De Poorter, E.; Hoebeke, J.: Efficient vertical handover in heterogeneous low-power wide-area networks, IEEE Internet of Things Journal, 7(3), 2019, 1960-1973. https://doi.org/10.1109/JIOT.2019.2961950
[9] Palmieri, F.: New energy-optimization challenges in the next-generation internet ecosystem, Information Sciences, 476, 2019, 373-374. https://doi.org/10.1016/j.ins.2018.10.042

[10] Pan, L.; Duan, Y.; Zhang, Y.; Xie, B.; Zhang, R.: A lightweight algorithm based on YOLOv5 for relative position detection of hydraulic support at coal mining faces, Journal of Real-Time Image Processing, 20(2), 2023, 1-12. https://doi.org/10.1007/s11554-023-01292-w
[11] Patnaik, S.; Nayak, M.; Viswavandya, M.: Strategic integration of battery energy storage and photovoltaic at low voltage level considering multiobjective cost-benefit, Turkish Journal of Electrical Engineering and Computer Sciences, 30(4), 2022, 1600-1620. https://doi.org/10.55730/1300-0632.3868
[12] Petrakis, E. G.; Sotiriadis, S.; Soultanopoulos, T.; Renta, P. T.; Buyya, R.; Bessis, N.: Internet of things as a service (itaas): Challenges and solutions for the management of sensor data on the cloud and the fog, Internet of Things, 3, 2018, 156-174. https://doi.org/10.1016/j.iot.2018.09.009
[13] Saeedian, M.; Adabi, M. E.; Hosseini, S. M.; Adabi, J.; Pouresmaeil, E.: A novel step-up single source multi-level inverter: Topology, operating principle, and modulation, IEEE Transactions on Power Electronics, 34(4), 2018, 3269-3282. https://doi.org/10.1109/TPEL.2018.2848359
[14] Thomas, A. C. B.; Rajanbabu, S.: Improving power quality by compensating voltage fluctuation in microgrid using golden section optimization algorithm, Technology, 12(2), 2021, 1-10.
[15] Wang, W.; Ma, B.; Hua, X.; Zou, B.; Zhang, L.; Yu, H.; Liu, X.: End-Cloud Collaboration Approach for State-of-Charge Estimation in Lithium Batteries Using CNN-LSTM and UKF, Batteries, 9(2), 2023, 114. https://doi.org/10.3390/batteries9020114
[16] Watanabe, M.; Takahashi, R.; Matsuda, K.; Seto, T.: A Cooperative Control Algorithm for Multiple SVRs Using Correlation of Measurement Data of Distribution Line, Electrical Engineering in Japan, 202(1), 2018, 3-10. https://doi.org/10.1002/eej.22964
[17] Wei, J.; Han, J.; Cao, S.: Satellite IoT edge intelligent computing: A research on architecture, Electronics, 8(11), 2019, 1247. https://doi.org/10.3390/electronics8111247
[18] Wu, Y.; Liu, Q.; Chen, R.; Li, C.; Peng, Z.: A group recommendation system of network document resource based on knowledge graph and LSTM in edge computing, Security and Communication Networks, 2020, 2020, 1-11. https://doi.org/10.1155/2020/8843803
[19] Yu, Q.; Wang, M.; Zhou, H.; Ni, J.; Chen, J.; Céspedes, S.: Guest editorial special issue on cybertwin-driven 6G: architectures, methods, and applications, IEEE Internet of Things Journal, 8(22), 2021, 16191-16194. https://doi.org/10.1109/JIOT.2021.3115295
[20] Zhang, Y.; Lyu, F.; Yang, P.; Wu, W.; Gao, J.: IoT intelligence empowered by end-edge-cloud orchestration, China Communications, 19(7), 2022, 152-156. https://doi.org/10.23919/JCC.2022.9837843