Research on Key Technologies of End-Side Computing Power Network Based on Field-Level Artificial Intelligence Reasoning for Terminal Equipment

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

[1,2,3,4]China Mobile Group Device Co., Ltd., Beijing, 100053, China
[1]nimaoch@163.com,[2]zhouting@cmdc.chinamobile.com,

[3]wanghengjiang@cmdc.chinamobile.com,[4]cuifang@cmdc.chinamobile.com

Corresponding author: Mao Ni, nimaoch@163.com

Abstract. To improve the end-side computing power of terminal equipment and promote its intelligence, it is necessary to improve the computing power combined with field-level AI reasoning technology. This paper focuses on the optimal deployment of the service function chain under cloud and fog collaborative architecture. Aiming at the problem that the traditional deployment scheme in a cloud computing environment makes it challenging to handle delay-sensitive requests, a corresponding solution is designed, and a complete service function chain deployment mechanism is realized. This mechanism is closely integrated with an intelligent identification network, which uses network function virtualization technology to decouple service functions and hardware devices. Moreover, this paper uses data simulation experiments to verify the computing power of the proposed model. From the experimental analysis, the improvement effect of the model presented in this paper is remarkable.

Keywords: terminal equipment; field-level; Artificial Intelligence reasoning; end-side computing power
DOI: https://doi.org/10.14733/cadaps.2024.S20.39-53

1 INTRODUCTION

Due to the lack of flexibility in traditional network architecture, deploying dedicated hardware devices (such as firewalls, deep packet detection, etc.) is necessary to provide services. Device providers provide these reliable hardware devices to service providers with specific hardware platforms and software services. Therefore, once they are deployed, the network functions are tightly coupled with dedicated hardware devices, the scalability is limited, and the subsequent integration will be subject to the device provider [1].

To deploy the updated solution, it is necessary to physically modify the hardware, which brings the most significant disadvantages of long device update times, increased operating costs, and complex management [10]. In addition, internet services are provided by multiple service providers, and modifying or adopting a new architecture requires consensus among relevant service providers, which is extremely difficult and can seriously hinder the innovation of the network. In this context, network virtualization has become one of the hot topics of expert research to meet the needs of future networks. Network virtualization can abstract and integrate the underlying resources of the network, forming a unified resource pool[16]. Software is used to achieve changes in network requests, allocate reasonably available resources externally, improve resource utilization, and meet resource sharing. This enables a physical network to support multiple virtual networks with different topologies. Due to the development of virtualization technology and the rapid expansion of network services, Software Defined Networks (SDN) and network function virtualization (NFV) have become technologies with excellent development prospects, and their flexible and efficient characteristics meet the network support requirements under the current environment[3]. Each network element device in traditional networks implements two functions: logical control and data forwarding. Each network element device exchanges information through protocols to obtain network topology and plan forwarding paths[4].

The emergence of low-power, wide-area Internet of Things technology enables IoT devices to be directly connected to corresponding base stations through LoRa, NB IoT, or eMCT technology, achieving long-distance communication. Compared to traditional Wi-Fi, Bluetooth, and ZigBee, low-power, wide-area Internet of Things has more advantages. However, in the current wireless vast area of the Internet of Things, most terminal devices can only be connected to the network through one of LoRa, NB IoT, eMCT, or GPRS technologies, which makes the network access method for IoT terminals single, causing mobile IoT services to be unable to switch networks based on business characteristics [19] dynamically. However, in the mobile Internet of Things context, wireless networks with different communication technologies coexist for a long time, and various wireless networks overlap and blend to form heterogeneous wireless networks. To leverage the characteristics and advantages of other communication technologies and switch in real-time according to user needs, designing an intelligent heterogeneous network fusion solution has become a public demand [13]. However, because different international institutions and organizations develop different types of wireless communication networks, it isn't easy to integrate various wide-area communication networks. Achieving heterogeneous network integration has become an essential topic in communication technology research. Currently, there are many solutions for heterogeneous network integration. The current practical scheme uses heterogeneous network integrated access terminals in different wireless networks to realize the interconnection between wireless wide-area networks[18]. When- heterogeneous network convergence is used to recognize the interconnection between wireless wide area networks, how to ensure that IoT devices are always in the optimal network has become a key research topic, which means that under the condition of multiple network coverage, the optimal network should be dynamically selected for access according to the terminal characteristics, network characteristics, business characteristics, and user preferences in heterogeneous networks so that wireless resources can be used more efficiently. Completely stable access to devices in mobile IoT scenarios [5]

Literature [2] proposes a decision model based on the Markov chain to judge the network state of the next moment to determine the best access network for users. Literature [9] proposes a decision model based on fuzzy neural networks that adaptively adjusts network selection parameters through output errors and continuously learns to meet the needs of users and services. Literature [7] proposes a network selection algorithm based on a utility function model. Balancing the price and network parameters allows users to allocate the best resources while the operator improves the maximum revenue. The literature [8] proposes a network selection algorithm based on the game theory model that can obtain the best price and bandwidth strategies through the game theory

between the user and the operator or choose multi-attribute network selection algorithms. Reference [12] proposes a low-power comprehensive gateway for the vast area of the Internet of Things that uses LoRa, NB IoT, and eMTC methods for network selection and connection. By designing a multi-attribute decision model that combines multilevel analysis and approximate ideal solution sorting, network selection for the vast area of the Internet of Things in a single scenario is carried out. Reference [14] also proposed combining two multi-attribute decision-making methods for heterogeneous wireless network selection between UMTS, WiMAX, and WLAN networks. Faced with the rapid development of the Wide Area Internet of Things, there have also been many studies on heterogeneous communication network fusion both domestically and internationally, mainly focusing on data fusion on the server side, communication fusion at the gateway base station, or network fusion on the terminal side.

Network awareness technology is one of the important research directions in networks under the current Internet [6]. Obtaining underlying network topology information through the application layer takes time and effort to achieve network awareness and application. Designing a logical network structure based on the application layer and physical network, which can perceive and obtain underlying details, not only plays a buffering role in getting the topology information of underlying network nodes but also reduces network resource consumption during transmission. It provides more efficient support for uploading information on the underlying physical state of the network and obtaining information on the upper application layer. In the research of network awareness methods, the application of network coordinate systems is one of the typical methods. The basic idea is to map the network into geometric space, where the nodes correspond to the coordinate position nodes in space. By one-to-one mapping, the logical space coordinates are converted into specific network coordinates when calculating the actual node coordinates of the network. This coordinate method has been widely applied to obtain network node information [20]. The idea of the virtual coordinate method [17] is to calculate the coordinates of network topology nodes through the relative coordinates in the logical matrix mapped by network nodes. The relative coordinates are obtained through the distance between spatial coordinate nodes. Use vectors to label and calculate the distance between nodes, and the vector information is recorded using binary sets, where the dimension information of the vector represents the number of current coordinate nodes. Vivaldi [15] is a lightweight and convenient method for calculating network coordinates. The distance between the coordinate nodes in the network is used to record their transmission delay, and the coordinate distance of the current node can be calculated by using the delay size of its adjacent nodes. Correspondingly, when the distance between two nodes is known, the distance can also be used to infer the delay between nodes. During the entire process, only a small amount of node information is needed to obtain the connection status of the whole network. This method can be easily and quickly deployed for large-scale network transmission.

Similarly, using vector information between nodes, the distance between nodes is obtained by setting input and output vectors for each node and calculating the inner product of the two vectors. The premise of this method is that the node vector information is linearly correlated, and when the correlation decreases, the measurement accuracy of this method will also decrease. Using RTT as a metric condition, the load-based network perception coordinates method constructs a perception network structure by utilizing node perception information (distance, delay, etc.) and link load. This method not only achieves network awareness to obtain information but also meets its requirements for accuracy and efficiency. The non-coordinate system method generally measures the distance information of some nodes in the network directly and then predicts the distance information of other nodes based on this information. This method does not use any coordinate information in the calculation process. The IDMaps [11] algorithm sets up tracking nodes in the network, applies them to network node information detection, uses network nodes and tracking nodes to detect distance from each other, and stores the distance information in the network node server. The distance between server nodes is calculated as follows: based on the information of the tracking node, the

distance between two servers is equal to the sum of the distances from each server node to its nearest tracking node.

This paper proposes an end-side computing power network model based on field-level AI reasoning for terminal devices, and the algorithm is improved according to the artificial intelligence network management requirements of terminal devices to strengthen the end-side computing power quality.

2 DESIGN OF SERVICE CHAIN DEPLOYMENT MECHANISM BASED ON CLOUD AND FOG COLLABORATION

2.1 Overall Design of Deployment Mechanism

This research is based on the intelligent integration identifier network architecture's three-layer and three-domain network architecture. The service function chain deployment mechanism proposed in this paper is closely integrated with the intelligent integration identifier network. As shown in Figure 1, this mechanism is mainly designed and implemented based on the three-layer mentioned above and three-domain smart integration identifier network architecture.

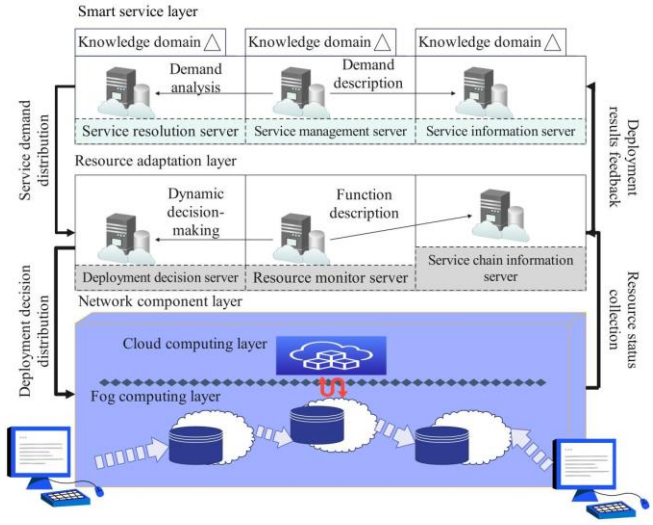


Figure 1: System design architecture.

The overall module design of the service chain deployment mechanism system under cloud and fog collaboration is shown in Figure 2. Based on the previous system analysis, the deployment of the service function chain needs to analyze service requirements, collect and store resource and service context information, calculate deployment decisions, and map the service function chain to the physical network.

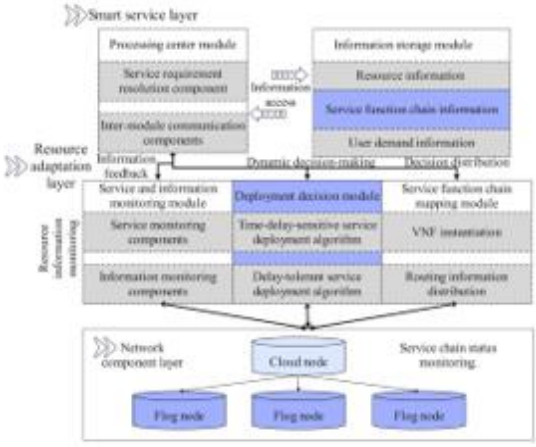


Figure 2: System module design.

The service demand analysis component in the processing center module, which realizes service request analysis, is shown in Figure 3. It can be seen that the processing center module faces the user terminal and receives the service demand sent by the user, that is, the demand information of the service function chain. However, this unprocessed information only includes the essential details constituting a service function chain, which can not be optimized for deployment. Therefore, the service request parsing component must process and analyze this information in fine granularity, extract the practical information, and transform it into the configuration file input into the deployment decision module.

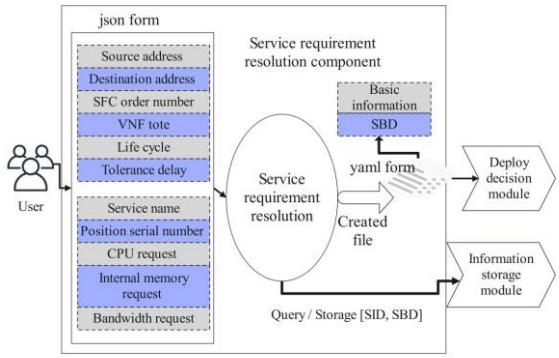


Figure 3: The parsing component of service requirement.

After the parsing process is completed, the parsing component of the service requirement generates the service identification SID according to formula (1); S_{type} represents the service type, which is the service chain based on cloud and fog collaboration in this paper, S_{name} represents the service name, which is the SFC serial number in this paper, $\phi 0$ represents the service identification generation function. The finally generated SID is then used to query whether a corresponding SBD exists.

$$SID \triangleq \phi(S_{type}, S_{name}) \tag{1}$$

The service requirements information is eventually converted into deployment files in YAML data format. In addition to supporting the basic information of Kuberneg behavior, a service behavior description SBD is generated according to formula (1) after analysis. SBD's topology behavior description includes the service request source address, b_S^{ST}, request destination address , etc. The performance behavior description has service delay requirements b_D^{SP}, bandwidth requirements b_B^{SP}, computing resource requirements b_P^{SP}, memory resource requirements b_M^{SP}, etc. The function behavior description information includes the service function name b_N^{SP} and service life cycle b_L^{SF}, etc.

$$SBD \triangleq \begin{bmatrix} \{b_S^{ST}, b_T^{ST}, \ldots\}_T \\ \{b_D^{SP}, b_B^{SP}, b_P^{SP}, b_M^{SP}, \ldots\}_P \\ \{b_N^{SF}, b_L^{SF}, \ldots\}_F \end{bmatrix} \tag{2}$$

The processing center module can query and update all the data in the information storage module and open the corresponding interface for other modules to operate.

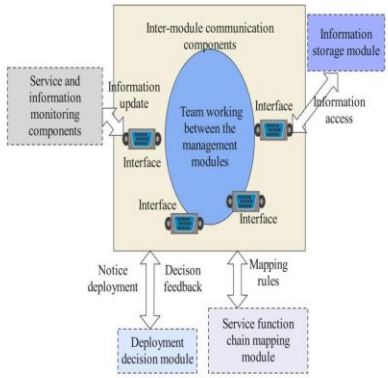


Figure 4: Communication components between modules.

2.2 Design of Service and Information Monitoring Module

Due to the differences in resources between cloud and fog nodes, the cloud and fog collaborative architecture must have the ability to monitor the resources of the whole network in real time to provide adequate data support for the deployment decision module. The main functions of the service and monitoring module in this system include collecting resource information and monitoring service information.

Figure 5 shows the information monitoring component's design in the resource adaptation layer. Firstly, information is collected in the network component layer. In collecting network status information, it is necessary to dynamically perceive the network topology and know the relationship between the location of network nodes and various devices in the network. At the same time, the network performance test tool qperf is used to obtain the link bandwidth and delay in the network, and qperf can use TCP and UDP to measure the network condition and use two nodes on a link as servers to communicate with clients and test them.

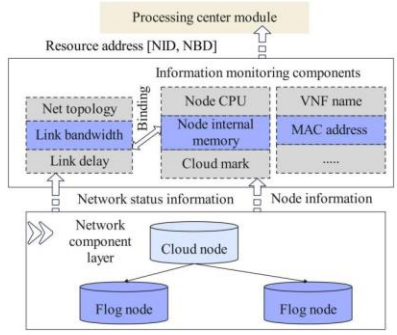


Figure 5: Information monitoring component.

After completing the information collection process, the information monitoring component will generate a component identification NID and a component behavior description NBD according to formulas (3) and (4). In formula (3), N_{type} represents the type of network component, namely the cloud or fog node, N_{device} represents the equipment information of the network component, and $\omega()$represents the generation function of component identification. In Formula (4), subscripts T, P, and F represent topology behavior, performance behavior, and function behavior, respectively. Finally, the generated key value pairs of NID and NBD and the collected resource information are returned to the processing center module for processing and saving.

$$NID \triangleq \omega(N_{type}, N_{device}) \tag{3}$$

$$NBD \triangleq \begin{bmatrix} \{b_L^{NT}, b_R^{NT}, \ldots\}_T \\ \{b_B^{NP}, b_D^{NP}, b_P^{NP}, b_M^{NP}, \ldots\}_P \\ \{b_N^{NF}, b_T^{NF}, \ldots\}_F \end{bmatrix} \tag{4}$$

2. Service monitoring component

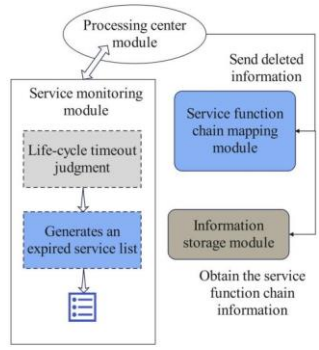


Figure 6: Service monitoring component.

Figure 6 shows the design of the service monitoring component. The service monitoring function must be completed with the processing center module, the information storage module, and the service function chain mapping module.

2.3 Design of Deployment Decision Module

The deployment decision module decides how to deploy the service function chain in the network. The two algorithms included in the deployment decision module can give optimization strategies for users' delay-sensitive or delay-tolerant service requests, respectively. At the same time, the deployment decision module needs to obtain the necessary information as algorithm input and feedback on the output results.

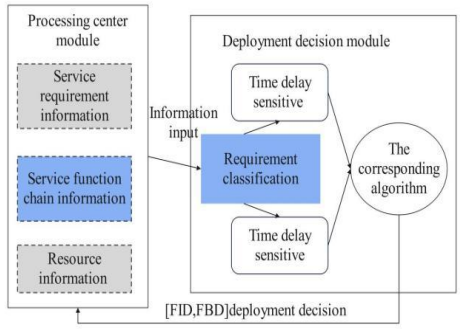


Figure 7: Design of deployment decision module.

As shown in Figure 7, the deployment decision module mainly interacts with the central processing module, which analyzes the user service demand information, including service function chain serial number, service function type, requested network and node resource information, etc., and sends it to the deployment decision module together with the current network resource information (network topology, link bandwidth, delay, etc.), node resource information (node CPU, memory, etc.) and service function chain related information (VNF information on nodes, etc.) collected by the service and information monitoring module.

If the deployment strategy is successfully generated, the synchronously generated group identification FID and the corresponding group behavior description FBD are returned to the processing center module. In Formula (5) for developing the FID, F_T represents the topological behavior of the group, F_P represents performance behavior, F_F represents function behavior, and $\varphi()$is the identification generation function of the group. In this paper, the FID is generated according to the contents of the FBD. In Formula (6) for developing FBD, subscripts T, P, and F respectively represent topology behavior, performance behavior, and function behavior, and topology behavior description information includes service function deployment location b_L^{FT} and service function link routing pathb_R^{FT}, performance behavior description information includes bandwidth performanceb_B^{FP}, delay performance b_D^{FP} and node cost performance b_C^{FP}, and function behavior description information includes component type $b_T^{FF}$and service function chain type b_S^{FF}, etc.

$$FID \triangleq \varphi(F_T, F_P, F_F) \tag{5}$$

$$FBD \triangleq \begin{bmatrix} \{b_L^{FT}, b_R^{FT}, \ldots\}_T \\ \{b_B^{FP}, b_D^{FP}, b_C^{FP}, \ldots\}_P \\ \{b_T^{FF}, b_S^{FF}, \ldots\}_F \end{bmatrix} \tag{6}$$

The physical network architecture based on this study is shown in Figure 8, which consists of the bottom user terminal network, two distributed fog computing layers, and the top centralized cloud computing layer. The lowest layer of the network is composed of user terminals, including mobile phones, Internet of Things terminals, and other devices. Service requests are generated by the user terminals at the bottom layer, which can be sent to the central fog node at the second layer. The primary fog node is connected with the auxiliary fog node at the third layer and finally to the cloud node at the fourth layer. Among them, the central fog node is closer to the user terminal than the auxiliary fog node and has a lower network delay, but at the same time, the resources are limited. The additional fog node has a larger resource pool, and the data center in the cloud has sufficient resources and high scalability. However, because it is far from the edge of the network, the long path of the deployed service function chain will increase the transmission delay, significantly increasing the overall end-to-end delay of the service. At the same time, due to the long routing path, it will also increase the consumption of network bandwidth resources. Therefore, the service function chain deployment mechanism based on cloud and fog collaborative architecture designed in this study can better handle delay-sensitive service requests by reasonably scheduling and deploying service functions in cloud and fog. At the same time, it can efficiently configure required resources for delay-tolerant service requests without strict delay constraints.

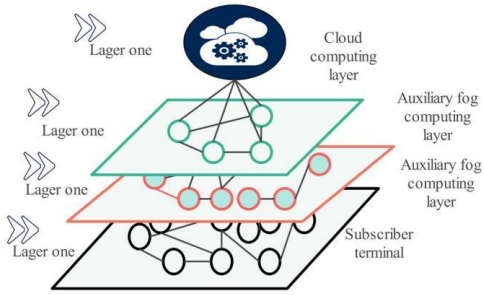


Figure 8: The architecture of a multi-layer cloud and fog collaborative system.

The multi-layer cloud and fog collaborative architecture proposed in this study can be modeled as an undirected weighted graph $G = (\mathrm{N}, \mathrm{E})$, in which $N = \{n_1^i, n_2^i, \ldots, n_j^i\}$ is defined as the collection of all general physical nodes in the network, n_j^i represents the j-th physical node in the i-th layer of the network, i=2,3 represents the fog computing layer of the second and third layers, and i=4 represents the cloud computing layer of the fourth layer. $E = \{e_{11}^{ik}, e_{12}^{ik}, \ldots, e_{jl}^{ik}\}$ is defined as a set of physical links connecting all physical nodes in the network, e_{jl}^{jk} represents the physical link between the j-th physical node of the i-th layer of the network and the l-th node of the k-th layer of the network. $c(n_j^i)$、$m(n_j^i)$ respectively represent the computing resources and memory resources provided on the noden_j^i, and $b(e_{jl}^{ik})$、$d(e_{jl}^{jk})$ respectively represent the bandwidth resources and delay of the link e_{jl}^{jk}.

At the same time, SFC requests with end-to-end delay constraints can be modeled as six tuples$req = <src, dst, b_r(e_{mn}), d_r, F_r, L_r>$, where src and dst are user terminals located at the first layer as source hosts and destination hosts, respectively, $F_r = \{f_1, f_2, \ldots, f_n\}$ represents an ordered set of all service

functions in the requested service function chain. e_{mn} represents the logical link between the service function f_m and f_n in the service chain request, and $b_r(e_{\mathrm{mn}})$ represents the bandwidth resource applied by the logical linke_{mn}. d_r is the maximum end-to-end delay that the service requested by the user can tolerate. L_r represents the life cycle of the ordered service function chain.

The total delay considered by the deployment algorithm for delay-sensitive services in this section is the sum of the above two delays, as shown in formula (7).

$$D(r) = \sum_{r\in R} \left(D_{pr}(r) + D_{prop}(r)\right) \tag{7}$$

$D_{pr}(r)$ is the total processing delay of the service function chain, as shown in formula (8), where$\Delta_f(n)$ represents the processing delay of the service function f on the cloud node or fog node, X_{fj}^i is the constraint of whether the service function f is deployed at the j-th node of the i-layer network, as shown in formula (9). $D_{prop}(r)$ represents the total link delay of a service function chain, as shown in formula (10), where $d_r(e_{mn})$ means the delay of the link between the service functions m and n, Y_{mn}^i is a constraint on whether the link i is deployed on the physical link between the service operates m and n, as shown in formula (11).

$$D_{pr}(r) = \sum_{n\in N} \sum_{f\in F_r} \Delta_f(n) \cdot X_{fj}^i \tag{8}$$

$$X_{fj}^i = \begin{cases} 1 \text{ service function f Deployed on page i Layer network layer j On nodes} \\ 0, \text{Other situations} \end{cases} \tag{9}$$

$$D_{prop}(r) = \sum_{e\in E} \sum_{m,n\in F_r} d_r(e_{mn}) \cdot Y_{mn}^i \tag{10}$$

$$Y_{mn}^i = \begin{cases} 1, \text{Virtual-link i Deployed on the physical link between service functions m and n} \\ 0, \text{Other situations} \end{cases} \tag{11}$$

To sum up, we can get the optimization objectives of the linear programming model of the deployment algorithm for delay-sensitive services, as shown in formula (12).

$$min(D(r)) = min\left(\sum_{r\in\mathbb{R}} \left(\sum_{n\in\mathbb{N}} \sum_{f\in\mathbb{F}_r} \Delta_r(n) \cdot X_{f'j}^i + \sum_{e\in\mathbb{E}} \sum_{m,R\in\mathbb{F}_r} d_r(e_{mn}) \cdot Y_{nm}^i\right)\right) \tag{12}$$

In addition, the model needs to set the following constraints:

1. Physical resource constraints

The VNF in each SFC request will be mapped to a fog node or cloud node in the physical network, so it is necessary to ensure that the computing resources and memory resources required by the VNF instance mapped to the node will not exceed the material resources available on the node at this time, to avoid the request being discarded due to resource overload, as shown in formulas (13) and (14).

$$\sum_{r\in R} \sum_{f\in F_r} X_{fj}^i \cdot p_r(f_i) \le c\left(n_j^i\right), \forall n_j^i \in N \tag{13}$$

$$\sum_{r\in R} \sum_{f\in F_r} X_{fj}^i \cdot m_r(f_i) \le m(n_j^i), \forall n_j^i \in N \tag{14}$$

2. Link resource constraints

In the service function chain deployment process, it is necessary to instantiate VNFs on designated nodes and map virtual links connecting two VNFs to physical links in the network. Therefore, it is required to satisfy the link resource constraint defined by Formula (15).

$$\sum_{r\in R} \sum_{fmf_n\in F_r} Y_{mn}^i \cdot b_r(e_{mn}) \le b\left(e_{jl}^{ik}\right), \forall e_{jl}^{ik} \in E \tag{15}$$

The node cost overhead is defined in formula (16):

$$C(r) = \sum_{r \in R} \left(C_{cpu}(r) + C_{mem}(r)\right) \cdot X_{flj}^{i} \cdot T_{fj}^{i} \tag{16}$$

In formula (16), $C_{cpu}(r)$ represents the node computation cost caused by a service function chain request, as defined in formula (17), $p_r(f)$ is the computation resource requested by a service function in the service function chain, and $\lambda_{cpu}(n)$ is the computation cost coefficient, which represents the increased node cost per unit computation resource requested by the user, as defined in formula (18), and the computation cost coefficient of fog nodes is generally more significant than that of cloud nodes. Hence, the node cost of deploying VNF on fog nodes is also more critical.

$$C_{cpu}(r) = \sum_{n \in N} \sum_{f \in F_r} \lambda_{cpu}(n) \cdot p_r(f) \tag{17}$$

$$\lambda(n) = \frac{c_{total}(n)}{c_{av}(n) + \epsilon} \tag{18}$$

In formula (18), $c_{total}(n)$ represents the computational resources on node n are represented, $c_{av}(n)$ represents the computational resources available at this time on Node n are represented, and $\mathcal{E}$ is a decimal value to prevent division by 0. It can be seen that the more computing resources occupied on a node, the greater the cost coefficient. Because the resource pool of fog nodes is much smaller than that of cloud nodes, the cost coefficient increases faster with the deployment process.

In formula (16), $C_{mem}(r)$ is the memory cost, which is defined as (19), where $\lambda_{mem}(n)$ and $m_r(f)$ are the memory cost factor and the memory resource requested by the service function. Finally, T_{fj}^{i} represents the lifetime of the service function f deployed on the j-th node in the i-th layer network, and the calculation process is shown in formula (20). If there is a VNF multiplexing, T_{fj}^{i} takes the maximum value between the difference between the two and 1.

$$C_{mem}(r) = \sum_{n \in N} \sum_{f \in F_r} \lambda_{mem}(n) \cdot m_r(f) \tag{19}$$

$$T_{fj}^{i} = \begin{cases} T_{new}, \text{No demultiplexing} \\ max\{T_{new} - T_{old}, 1\}, \text{Multiplexing} \end{cases}, i, j \in N, f \in F_r \tag{20}$$

The constraints of the delay-tolerant request deployment model are as follows:

1. Physical resource constraints

The physical resource constraints of the deployment algorithm model for delay-tolerant services are the same as those defined in formulas (13) and (14) in the previous section.

2. Link resource constraints

The link resource constraints of the deployment algorithm model for delay-tolerant services are the same as those defined in formula (15) in the previous section.

3. Delay constraint

The optimization goal of the deployment algorithm model for delay-tolerant services is to reduce the node cost in the deployment process. However, the delay problem still needs to be considered. While lowering the node cost, the end-to-end delay of the service function chain cannot exceed the maximum tolerance delay of users, as defined in the Formula (21).

$$\sum_{e_{jl}^{ik} \in E} \sum_{f_m, f_n \in F_r} Y_{mn}^{i} \cdot d\left(e_{jl}^{ik}\right) \leq d_r \tag{21}$$

3 MODEL CONSTRUCTION AND VERIFICATION

Introducing virtualization technology into the WSN network can effectively alleviate the problems of low resource utilization and poor service flexibility in the WSN network. Meanwhile, using the idea of virtualization, all sensor nodes in a given area can be abstracted into a single shared infrastructure, giving full play to the advantages of resource sharing. According to different QoS of users, when multiple tasks are executed concurrently, sensor nodes are flexibly grouped and used to form different VSNs to maximize the multiplexing of sensor resources and significantly improve the flexibility and scalability of the WSN. VSN with resource requirements is called VSNR, which includes virtual sensor node resource requirements and virtual link resource requirements. VSNR needs to be mapped to the underlying physical sensor network resources to provide services for users; that is, virtual sensor nodes and virtual links are mapped to the underlying physical sensor nodes and physical paths that meet their conditions. However, there is a competitive relationship among different VSNSPs, and each VSNSP cannot know each other's resource requirements in resource competition, so the resource competition process among multiple VSNSPs belongs to the dynamic game process with incomplete information. Virtualization can support the active sharing of physical resources, and at the same time, units and different VSNRs can be mapped to the same sensor nodes and physical paths; that is, some physical resources need to be shared when multitasks are executed concurrently. Therefore, allocating resources to VSNRs with the exact resource requirements to maximize the interests of both resource supply and demand sides is very important for improving the utilization rate of network resources and satisfying the user service experience. Combined with the algorithm model in the second part, this paper proposes an end-side computing power network model based on field-level AI reasoning for terminal devices, as shown in Figure 9.

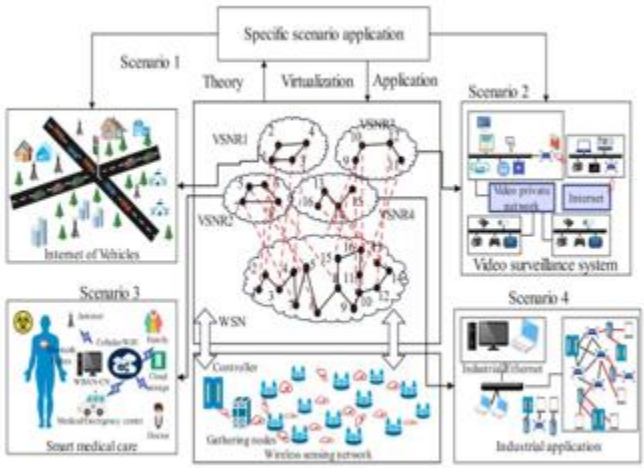
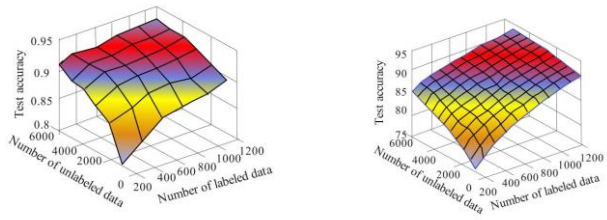


Figure 9: End-side computing power network model based on field-level AI reasoning for terminal equipment.

To verify whether the end-to-side computing power network model based on field-level AI reasoning for terminal devices can effectively improve the computing power of the system and the capability of the terminal system, this paper mainly carries out the computing power and timely response effect of the terminal AI driving system by simulating driving pictures and video recognition. More than 100,000 data graphs are collected in the CIFARl0 data set, and the simulation analysis is carried out through the system in this paper. At the same time, each vehicle terminal uses training images

to participate in Semi-VFL tasks. We set the batch size as B=64, the number of local iterations as E=5, and the number of global iterations (communication rounds) as 100. Random gradient descent is used for the optimizer, and the learning rate is set to 0.01; the specific threshold is u=0. 95, and the trade-off parameters are set to v=1 and g=10. The result is shown in Figure 10.



(a)Experiments results (b)Fitting results

Figure 10: Experimental and fitting results.

Figure 10 shows the experimental and fitting results for the CIFARl0 data set. Figure 10 (a) shows the relationship between test accuracy, the amount of marked data, and the amount of unmarked data. With the increase in unlabeled data, the test accuracy increases. Moreover, the increase in the number of labeled data points can also improve the test accuracy of the global model. At the same time, from Figure 10 (b), we can see that the fitting results are excellent. On this basis, the model proposed in this paper is verified, and the evaluation results shown in Table 1 are finally obtained through the verification of multiple groups of data.

Number	*Data evaluation*	*Number*	*Data evaluation*	*Number*	*Data evaluation*
1	*88.13*	*15*	*88.90*	*29*	*85.31*
2	*82.19*	*16*	*84.88*	*30*	*88.65*
3	*90.18*	*17*	*87.54*	*31*	*83.63*
4	*89.70*	*18*	*82.77*	*32*	*89.10*
5	*87.11*	*19*	*87.25*	*33*	*81.62*
6	*85.60*	*20*	*81.15*	*34*	*83.12*
7	*84.96*	*21*	*83.56*	*35*	*85.36*
8	*80.42*	*22*	*82.09*	*36*	*82.61*
9	*87.56*	*23*	*90.10*	*37*	*81.85*
10	*90.66*	*24*	*81.46*	*38*	*89.74*
11	*89.40*	*25*	*85.80*	*39*	*88.16*
12	*81.53*	*26*	*86.72*	*40*	*84.61*
13	*84.25*	*27*	*88.85*	*41*	*85.27*
14	*80.16*	*28*	*80.36*	*42*	*83.23*

Table 1: Verification results of model data.

From the above data analysis, the end-side computing power network model based on field-level AI reasoning for terminal equipment proposed in this paper has a specific improvement effect.

4 CONCLUSIONS

In the intelligent identification network, Network Function Virtualization (NFV) technology can be used to migrate network functions from special hardware devices to software-based applications, which realizes the decoupling of network functions and hardware devices and enables Virtual Network Function (VNF) to be instantiated in different locations of the network. In addition, Service Function Chaining (SFC) can use NFV technology to improve flexibility and on-demand deployment capability. SFC links VNFs together according to business logic requirements, and these VNFs are usually deployed on many computing nodes. Network traffic traverses each VNF through virtual links between VNFs to obtain the required network services. An optimized deployment mechanism for service function chains is proposed to solve the problem of resource difference and network overhead encountered in deploying service function chains in cloud and mist collaborative architectures. For different service requirements of delay sensitivity or delay tolerance, different deployment scheduling algorithms are designed, deployment nodes and routing links are intelligently selected in cloud and fog collaborative architecture, and an efficient and state-aware deployment mechanism of the service function chain is realized to meet service requirements and improve service acceptance rate and resource utilization rate. According to the test data analysis, the end-side computing power network model based on field-level AI reasoning for terminal equipment proposed in this paper has a specific improvement effect.

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

REFERENCE

[1] Chen, L.; Jiang, D.; Song, H.; Wang, P.; Bao, R.; Zhang, K.; Li, Y.: A lightweight end-side user experience data collection system for quality evaluation of multimedia communications, IEEE Access, 6, 2018, 15408-15419. https://doi.org/10.1109/ACCESS.2018.2794354
[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] Eshratifar, A. E.; Abrishami, M. S.; Pedram, M.: JointDNN: An efficient training and inference engine for intelligent mobile cloud computing services, IEEE Transactions on Mobile Computing, 20(2), 2019, 565-576. https://doi.org/10.1109/TMC.2019.2947893
[4] He, P.; Zhou, Y.; Duan, S.; Hu, X.: Memristive Residual CapsNet: A hardware friendly multilevel capsule network, Neurocomputing, 496, 2022, 1-10. https://doi.org/10.1016/j.neucom.2022.04.088
[5] Ji, W.; Liang, B.; Wang, Y.; Qiu, R.; Yang, Z.: Crowd V-IoE: visual Internet of everything architecture in AI-driven fog computing, IEEE Wireless Communications, 27(2), 2020, 51-57. https://doi.org/10.1109/MWC.001.1900349
[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] 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
[8] 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
[9] 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

[10] Ruan, L.; Guo, S.; Qiu, X.; Meng, L.; Wu, S.; Buyya, R.: Edge in-network computing meets blockchain: A multi-domain heterogeneous resource trust management architecture, IEEE Network, 35(5), 2021, 50-57. https://doi.org/10.1109/MNET.110.2100029

[11] Saeedian, M.; Adabi, M. E.; Hosseini, S. M.; Adabi, J.; Pouresmaeil, E.: A novel step-up single source multilevel inverter: Topology, operating principle, and modulation, IEEE Transactions on Power Electronics, 34(4), 2018, 3269-3282. https://doi.org/10.1109/TPEL.2018.2848359

[12] 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

[13] Wang, Z.; He, S.; Li, Q.; Liu, B.; Razzaghi, R.; Paolone, M.; Rachidi, F.: A full-scale experimental validation of electromagnetic time reversal applied to locate disturbances in overhead power distribution lines, IEEE Transactions on Electromagnetic Compatibility, 60(5), 2018, 1562-1570. https://doi.org/10.1109/TEMC.2018.2793967

[14] 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

[15] 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

[16] Wu, X.; You, L.; Wu, R.; Zhang, Q.; Liang, K.: Management and Control of Load Clusters for Ancillary Services Using Internet of Electric Loads Based on Cloud–Edge–End Distributed Computing, IEEE Internet of Things Journal, 9(19), 2022, 18267-18279. https://doi.org/10.1109/JIOT.2022.3156954

[17] 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

[18] Yan, Y.; Liu, Y.; Fang, J.; Lu, Y.; Jiang, X.: Application status and development trends for intelligent perception of distribution network, High Voltage, 6(6), 2021, 938-954. https://doi.org/10.1049/hve2.12159

[19] Yang, Z.; Liang, B.; Ji, W.: An intelligent end–edgecloud architecture for visual IoT-assisted Healthcare systems, IEEE Internet of Things Journal, 8(23), 2021, 16779-16786. https://doi.org/10.1109/JIOT.2021.3052778

[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