Hindawi
Complexity
Volume 2021, Article ID 6455617, 11 pages
https://doi.org/10.1155/2021/6455617

Research Article

Machine Learning-Based Offloading Strategy for Lightweight User Mobile Edge Computing Tasks

Shuchen Zhou,[1,2] **Waqas Jadoon**,[2] **and Junaid Shuja**[2]

[1]*Institute of International Education, Huanghuai University, Zhumadian 463000, China*
[2]*Department of Computer Science, COMSATS University Islamabad, Abbottabad Campus, 22060 Islamabad, Pakistan*

Correspondence should be addressed to Shuchen Zhou; 20101192@huanghuai.edu.cn

Received 25 April 2021; Revised 28 May 2021; Accepted 29 May 2021; Published 15 June 2021

Academic Editor: Zhihan Lv

This paper presents an in-depth study and analysis of offloading strategies for lightweight user mobile edge computing tasks using a machine learning approach. Firstly, a scheme for multiuser frequency division multiplexing approach in mobile edge computing offloading is proposed, and a mixed-integer nonlinear optimization model for energy consumption minimization is developed. Then, based on the analysis of the concave-convex properties of this optimization model, this paper uses variable relaxation and nonconvex optimization theory to transform the problem into a convex optimization problem. Subsequently, two optimization algorithms are designed: for the relaxation optimization problem, an iterative optimization algorithm based on the Lagrange dual method is designed; based on the branch-and-bound integer programming method, the iterative optimization algorithm is used as the basic algorithm for each step of the operation, and a global optimization algorithm is designed for transmitting power allocation, computational offloading strategy, dynamic adjustment of local computing power, and receiving energy channel selection strategy. Finally, the simulation results verify that the scheduling strategy of the frequency division technique proposed in this paper has good energy consumption minimization performance in mobile edge computation offloading. Our model is highly efficient and has a high degree of accuracy. The anomaly detection method based on a decision tree combined with deep learning proposed in this paper, unlike traditional IoT attack detection methods, overcomes the drawbacks of rule-based security detection methods and enables them to adapt to both established and unknown hostile environments. Experimental results show that the attack detection system based on the model achieves good detection results in the detection of multiple attacks.

1. Introduction

Mobile edge computing and storage (MECC) technology, as a new computing and storage paradigm, deploys centralized cloud data centers in a distributed form on the side of the access network close to the data source, attempting to deeply integrate Internet Service Providers (ISPs), mobile operators, and IoT devices to perform many operations such as service awareness, data transmission, information processing, and control optimization near the data source. MECC technology can provide distributed computing and storage functions in wireless access networks close to mobile devices [1]. However, in the face of complex and diverse IoT application scenarios, the practical application of MECC technology still faces many challenges. On the one hand, for lightweight IoT devices, frequent storage and computation operations will consume a lot of energy, which is a great challenge for battery-powered devices only, and the development of battery technology still cannot meet the energy demand of IoT applications [2]. Therefore, without a new way to supply energy or improve energy efficiency, it will greatly limit the application of MECC in such scenarios. On the other hand, as an important part of IoT, the integration of telematics and MECC technology can better support the rapid implementation and application popularity of advanced assisted driving and autonomous driving in the future. Unlike stationary or low-speed mobile IoT devices, the high-speed mobility of smart vehicles, the transient interaction of information, and the dynamic topology of the network will bring great challenges to the application of

MECC technology in IoT [3]. Also, compared to resource-constrained IoT devices, vehicles with enhanced functionality can be considered as mobile edge infrastructure, enabling more ground-hugging computing and storage applications by providing their idle resources.

Mobile edge computing tasks mostly involve distributed messing tasks, so distributed computing is needed. As distributed computing, the cloud computing controller, when processing computing requests, generally assigns computing tasks to any available computing resource on the network to complete and then sends them back to the computing requestor after unified processing [4]. In this process, when the number of user requests is huge, the total amount of data is divided into a huge number of task requests, if not a huge amount, so the bandwidth (or channel capacity) occupied by data transmission on the network (wired and wireless) will become extremely large. Even though the bandwidth capacity of a single set of network equipment has now developed to more than THz, it cannot meet the huge number of concurrent tasks. For example, part of the time, no matter what kind of Internet access is available, it will become abnormally slow, which is deeply felt by people [5]. (1) A huge number of concurrent computing task requests, even if the individual task data volume is not large, will also bring the server overwhelmed; (2) the communication link of the service request is too long, often having to go through multiple routers forwarding, to reach the target server; (3) the operator of the router arrangement, not following the maximum demand to build, of course, cannot carry the burst of computing task requests [6]. The computing power of cloud computing is often sufficient, but the channel bandwidth or the number of channels is the real bottleneck that restricts cloud computing from handling burst computing tasks; that is, the actual situation is that the network is busy while the computing resources (such as CPU) are idle, which is like most of the reasons why personal computers become slow.

Based on this, the concept of computational offloading has arisen. Nowadays, it can be found that some terminal devices installed on the user side by telecom operators have taken the role of computing offloading tasks. Regarding computation offloading under cloud computing and mobile edge computing offloading, some results have been achieved on computation offloading delay strategies, energy consumption, and energy efficiency control strategies, and their direct combination of both, but there are still many urgent issues to be solved, which are the focus of this research paper. With the extension of mobile edge computing in composition to multiaccess edge computing, the addition of various heterogeneous networks makes the stability of the system a great challenge, and it is impossible to establish a completely reliable computation offloading model without transmission errors and failures occurring, so this paper introduces the failures during computation offloading into the research to make the computation offloading more robust. In addition, with the demand of energy consumption of mobile devices, this paper discusses the control strategy of using SWIPT technology in mobile edge computing, which provides more options for MEC to make accurate offloading decisions and efficient resource allocation; due to the many problems in the implementation of wireless energy-carrying communication, with the progress and development of multiple antenna technology of spatial diversity and multifrequency antenna technology, this paper prompts to adopt a similar frequency division multiplexing approach to open a new research path for wireless energy-carrying communication.

In this paper, we study the computational offloading problem of wireless energy-carrying communication under mobile edge computing. First, a dynamic offloading and resource scheduling optimization model for a multiuser mobile edge cloud SWIPT system is developed, and then the hybrid nonconvex optimization problem is transformed into a zero-parity gap optimization problem. Based on this, by solving the optimization problem, this paper obtains an iterative algorithm to obtain the optimal strategies for clock frequency control, transmission power allocation, offloading ratio, and received power split ratio, while achieving the minimization of the system energy consumption. Finally, the algorithm is verified by extensive simulations to show that the account has a good performance in terms of system energy consumption. Also, in the context of mobile edge computing, a SWIPT scheme with a multiuser frequency division multiplexing approach is proposed by analogy with different wireless communication schemes, and a system optimization model for minimizing energy consumption based on mobile edge computing offloading is established. Then, a mixed-integer nonlinear programming problem is transformed into a convex optimization problem based on a reasonable relaxation of optimization variables and solved in detail by a Lagrange dual method; based on this, an algorithm is proposed to apply an integer programming branch delimitation algorithm to solve the optimization problem based on the use of the dual method. Finally, the good performance of the frequency division SWIPT technique proposed in this paper for energy minimization scheduling strategy in mobile edge computing offload is demonstrated through simulation analysis and comparison.

2. Status of Research

Li et al. introduced energy harvesting techniques to MEC systems and proposed an energy-saving strategy considering latency and offloading failures [7]. Besides, Yousafzai et al. first proposed a multiuser multitasking mobile edge cloud computing offloading problem and proposed an optimal energy harvesting strategy based on the Lyapunov optimization method and greedy algorithm [8]. Xiao et al. derived an optimal policy consisting of CPU frequency and mobile device (MD) transmit power using the Lyapunov optimization method [9]. Cong et al. studied the joint load management and resource allocation problem in mobile edge cloud systems based on energy harvesting in small-cell networks and designed an algorithm to maximize the number of offloaded users [10]. However, the amount of energy harvested from the surrounding environment is still limited in practice because the amount of energy collected is susceptible to environmental or natural influences, which are uncontrollable around mobile devices [11]. Wireless

energy harvesting can provide more reliable energy than acquiring unstable and uncontrolled energy from the surrounding environment because it allows energy transmission under human control and is less affected by the natural environment. Hu et al. proposed a fog node delay-aware application module management strategy that consists of two parts: first, the placement of the application module policy and then the inactive idle resources to which the module is forwarded and reduces the process to a constraint-based optimization problem to solve [12].

Shen et al. proposed deep learning as a new intrusion detection technique in the IoT environment with good results [13]. They point out that thousands of zero-day attacks have emerged due to the inclusion of various protocols. These protocols are mainly from the IoT and most of them are small variants of previously known cyber attacks. This situation suggests that even advanced mechanisms like traditional machine learning have difficulty detecting these small attack variants over time. Cao et al. propose to use behavioral modeling to detect intrusions in smart homes by detecting anomalies associated with nonplaying roles (NPCs), i.e., playing different roles around and inside the smart home [14]. Although the proposed work does not disclose which IA or hardware implementation is used, it claims to enable cost-efficient and easily verifiable autonomous monitoring for intrusion detection. The IoT is designed as a network of small devices distributed over the Internet, and its scheme is designed using state estimation and sequential hypothesis testing techniques to address the limitations of existing research [15]. The main idea of the design is to exploit the high spatiotemporal correlation between successive observations in IoT environmental monitoring to predict future observations based on previous reviews [16]. They evaluated the security of the scheme by game theory analysis. The results show that the scheme can obtain good detection results even when aggregators launch FDI attacks with very low frequency and intensity [17].

3. Machine Learning Analysis of Offloading Strategies for Lightweight User Mobile Edge Computing Tasks

3.1. Lightweight User Mobile Edge Computing Task Offload Policy Design. Fog nodes are an important part of the cloud computing system, and their task scheduling and computing resource allocation are important factors that affect the quality of service and resource usage efficiency of the cloud computing system tasks. In practical application scenarios, the tasks generated by end devices and users not only have real-time requirements, but also have heterogeneity in the demand for computing resource allocation for different types of tasks (i.e., they can be classified into different task types according to the heterogeneity of the demand for computing resource allocation), and the limited computing resource capacity of fog nodes can hardly meet the computing resource allocation demand for different types of tasks at the same time [18]. Therefore, the balance of computing resource allocation between different types of tasks by fog node computing resource allocation algorithms is a key factor affecting the quality of task services and the efficiency of fog node computing resource usage. The balance of fog node computing resource allocation not only can avoid the problem of "resource starvation" for certain types of tasks, but also can ensure the optimization of fog node throughput.

Moreover, the limited capacity and computing power of the fog nodes make the tasks need to be queued and buffered within the fog nodes, so the order of task execution within the fog nodes also affects the quality of service of the tasks and the usage efficiency of the fog nodes' computing resources, which means that the fog nodes need to have not only a suitable task sequencing buffer scheduling algorithm to optimize the order of task execution within the fog nodes, but also means that the task scheduling and computational resource allocation algorithms are coupled with each other, which has an impact on the quality of service of tasks and the efficiency of using computational resources of fog nodes. Therefore, the task scheduling algorithm and the computational resource allocation algorithm of the fog node need to collaborate so that as many real-time tasks as possible can be processed before the deadline while ensuring balanced computational resource allocation and improving the throughput of the fog node. However, from the practical point of view of tasks, the randomness parameters (e.g., task type, deadline, and data size) and uncertainty (e.g., arrival time and execution time) of real-time heterogeneous tasks increase the difficulty of real-time heterogeneous tasks processing by fog nodes. In this case, the waiting time of a task in the task queue of a fog node will become uncertain [19]. Although task execution time can be predicted, inaccurate task execution time prediction increases the difficulty of optimizing task scheduling and computational resource allocation policies of fog nodes. The task processing process structure of the cloud computing system is shown in Figure 1, which includes the terminal layer, the fog computing layer, and the cloud computing layer.

As far as the purpose of computational offloading is concerned, it can be divided into offloading for performance enhancement and offloading for energy saving. As applications in mobile devices become more complex, it is difficult to ensure that tasks can be completed within the specified time constraints by performing computation only on the mobile device. To achieve this goal of real time, for example, navigation robots need to make and successfully evade obstacles before they encounter them. Applications such as driverless and intelligent transportation need to be supported by powerful computational processing power, and computational offloading solves this problem by offloading the heavy computational tasks to other devices. Similarly, as in the case of context-aware computing, the limited computing power of mobile devices greatly limits the spread of applications, and computation offloading will improve the overall computing power of mobile devices. To ensure that computation offloading is efficient and feasible, offloading does improve performance only if the time taken by the mobile device to complete the computation task locally is greater than the time consumed to transfer the

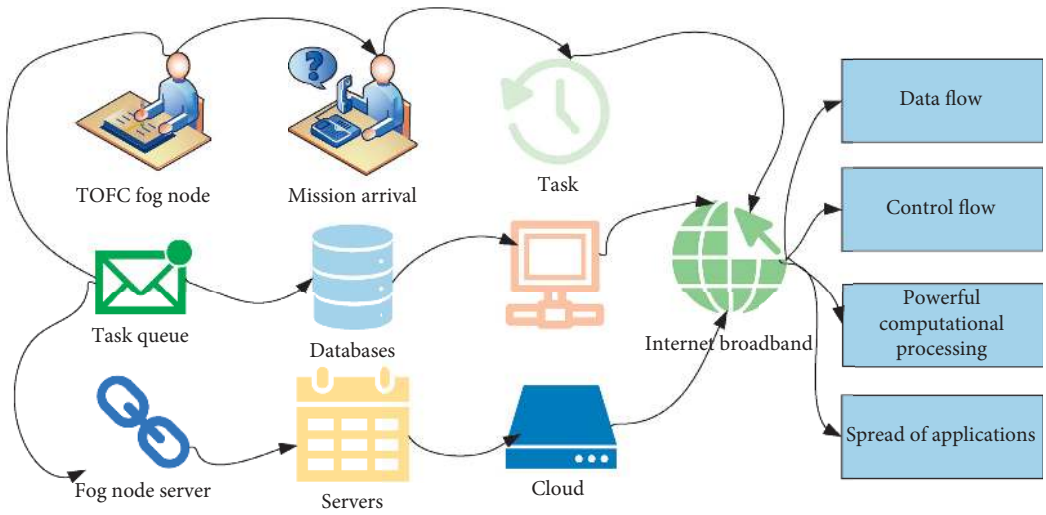

Figure 1: Schematic diagram of the task processing process structure.

computation data to the server. Therefore, the computational complexity of the task, the computational rate of the mobile device, the amount of data uploaded for the computational task, and the channel transmission rate all have a significant impact on the performance of computational offloading. Nowadays, although battery-making technology, fast charging technology, and wireless charging technology are developing rapidly, the limited battery capacity of mobile devices still cannot satisfy people's desire to explore various mobile applications. Energy saving is still a big issue. And computation offloading can reduce energy consumption on the mobile site by offloading energy-intensive computation tasks to the server side. At the same time, to ensure that offloading is energy-efficient and reliable, it is energy-efficient only when the energy consumed to complete the task on the mobile device is greater than the energy consumed to upload the offloaded data to the server for data transfer. Therefore, the computational task load, the amount of uploaded data, and the CPU frequency of the mobile device have a great impact on the energy consumption, as well as the transmission power of the mobile device to communicate with the server.

Full offload means that all computing tasks are offloaded and completed on the server or all are executed and completed on the local device; partial offload means that a part of computing tasks are completed on the mobile device (locally executed) and the rest are offloaded to the server for execution. As in Figure 2, the forms of dichotomous and partial offloading are briefly described. For some highly integrated or relatively simple tasks that cannot be further divided, such as applications with a directed acyclic graph DAG structure, as in Figure 2, they must be executed locally on the mobile device or offloaded to the server. The optimal allocation strategy is derived according to different optimization goals. Some of these subtasks can be optionally executed on the server side or locally on the mobile device. As in Figure 2, many applications, such as face recognition, augmented reality, virtual reality, speech recognition, etc., can be divided with finer granularity, e.g., the data are divided into arbitrary proportions at the bit level, for task offloading calculations to achieve certain goals.

To efficiently utilize the computational offloading technique, research work can be carried out in three aspects: minimizing energy consumption while satisfying computational latency constraints; minimizing computational latency; and jointly optimizing both energy consumption and computational latency. To minimize the energy consumption on the mobile device side while satisfying the computational latency constraint of the application, the energy consumption generated by local computation and data transfer will greatly affect the final resource allocation strategy. If the offloaded tasks do not need to be computed locally, the computation tasks that are offloaded to the server can save the energy consumption of the mobile device [19]. At the same time, the mobile device consumes energy by uploading the offloaded data to the server and receiving the computation results from the server side. The tradeoff between the energy consumption of local computation and the energy consumption of uploading and receiving data finally leads to the optimal resource allocation policy. The problem of minimizing local computation latency and the latency generated by computation offloading (including the upload, server computation, and return) is finally achieved by optimizing the local computation frequency, mobile device transmission power, server computation frequency, and mobile device reception power and then optimizing the offloading ratio.

This study is to achieve a compromise between energy consumption and execution time of mobile devices by weighting the energy consumption and computation latency in a multiuser and multichannel environment. The final optimization strategy of this study needs to consider the amount of computational task data per user, server and mobile device computational capacity, communication channel, and mobile device energy consumption to achieve a compromise between energy consumption and computational latency by jointly considering related factors. At the same time, by adjusting the weights of the two,

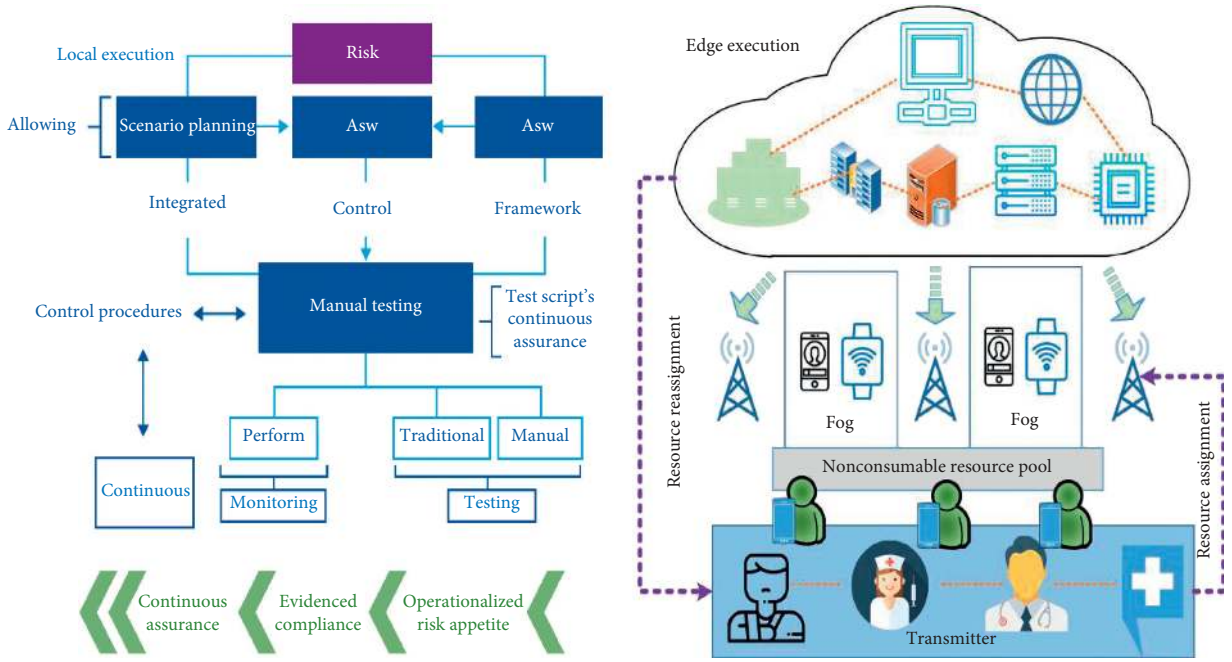

Figure 2: Full and partial uninstallation.

corresponding optimization strategies can be given for different requirements of different computing task types, such as latency-sensitive tasks and energy-consumption-sensitive tasks.

3.2. Machine Learning Algorithm Design. Any packet in dark network traffic data from an IoT device that does not meet either of the two criteria above is a configuration error or other traffic. A configuration error is a large duplicate traffic flow targeting one destination port. This communication traffic can occur for a variety of reasons, including misconfigured network address translation (NAT) rules and routing table errors. The cases of other traffic are rare and largely negligible. The misconfigured packet traffic is determined by a probabilistic model in this section. Combining the characteristics of IoT dark network traffic with the advantages of machine learning and IoT dark network attack node classification model based on dark network traffic and machine learning is proposed, as shown in Figure 3.

Empirical dark web data can help characterize Internet-scale malicious activity, but it still may not provide insight into the behavior of unsolicited IoT devices. Therefore, filtering of darknet sessions originating from IoT devices is necessary. In this chapter, this problem is addressed by correlating packet information from the CAIDA darknet database with data measurements of active IoT devices obtained from the Internet. Here the data of the active IoT devices are obtained using data from the Census and Shodan search engines. A key issue in correlating the two measurements is the need to properly clean the darknet data and filter out misconfigured traffic that is incorrectly directed to the darknet due to software, hardware, or routing errors.

SVM is a supervised machine learning algorithm for binary classification, usually used for classification problems. The idea of SVM classification is to find a hyperplane in a space where all sample points in the sample set have the shortest distance from the hyperplane and where the hyperplane can divide all samples. It creates the boundary by determining a two-dimensional boundary line through space. The hyperplane equation can be written in the following form:

$$W^T X = b. \tag{1}$$

Suppose that $P(x_1, x_2, \ldots, x_m)$ is a sample point and x_k is the k-th feature variable $P(x_1, x_2, \ldots, x_m)$. The formula for calculating the distance v to the hyperplane can be expressed as follows:

$$v = \frac{\left|w_1 \times x_1 + w_2 \times x_2 - w_3 \times x_3 + w_4 \times x_4 - \ldots + w_n \times x_n + b\right|}{\sqrt{w_1^2 + w_2^2 + \ldots + w_n^2}}. \tag{2}$$

For nonlinearly divisible support vector machines, to find the nonlinear hyperplane, a nonlinearly divisible dataset in low-dimensional space can be transformed into a linearly divisible dataset in high-dimensional space using a kernel function. As a weak differentiator SVM for attack node classification, the radial basis kernel function (RBF) is used

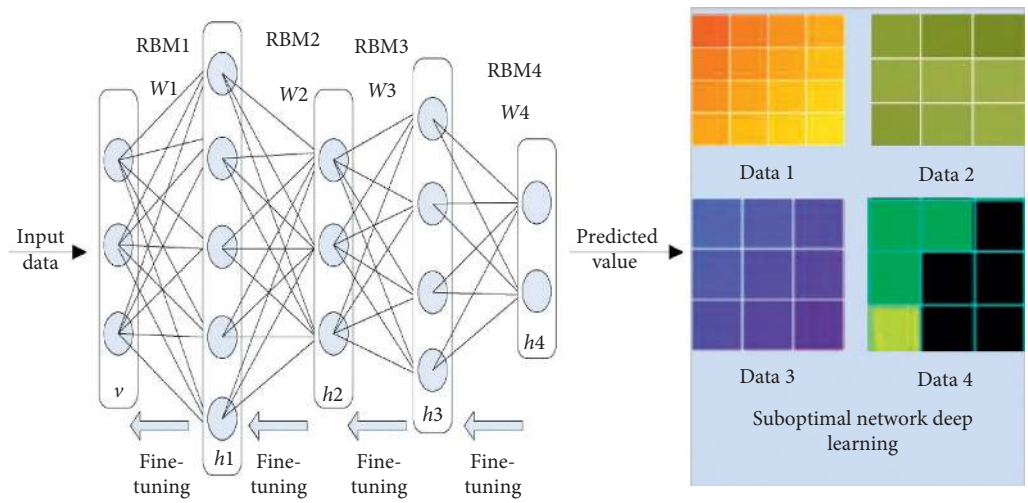

Figure 3: Machine learning-based IoT node classification model.

as its kernel function, also called Gaussian kernel function, which can be expressed as follows:

$$K(x, y) = \frac{e^{|x+y|^3}}{2\sigma^2}, \tag{3}$$

where σ is the arrival rate and $x + y$ is the distance from the sample point to the center. The optimization of a nonlinear separable support vector machine can be expressed as follows:

$$\max \frac{1}{2} W^T W + C \sum_i^m = 1 \tag{4}$$

$$\text{s.t. } y_i\left(W^T \times x_i\right) \geq 1 + \varepsilon, \tag{5}$$

where C is a constant greater than 0, W is the degree of penalty for incorrect samples, and $\varepsilon > 1$ is the slack variable. Constructing a Lagrange function for the solution, the optimization problem is transformed into the following equation:

$$\max \frac{1}{2} \sum_i^m \sum_{j=1}^m K\left(x_i, y_j\right) = 1 \tag{6}$$

$$\text{s.t. } \sum_i^m a_i y_i = 0. \tag{7}$$

Under the existing cloud computing system, the long-distance communication between IoT devices and cloud data centres cannot guarantee timely data transmission to cloud data centres due to the unstable nature of backhaul links. Besides, the entry of large-scale IoT devices can cause huge pressure on the access network [20]. On the other hand, the monolithic cloud computing model, which is far away from IoT devices, will lead to the convergence of IoT business traffic in the centralized cloud computing platform, which will significantly increase the computational pressure on the cloud data center and even result in the "collapse" of computing devices due to the data flood. Therefore, how to design a new computing model for ultrahigh device connectivity and traffic density is a major challenge for future IoT applications.

$$b = \limsup \frac{1}{M} \sum_{t=0}^{M} E\|b(t)\|. \tag{8}$$

To meet the growing number of device connections and traffic density, ISPs can meet this challenge by deploying more data centres. However, this approach will significantly increase the cost of processing data for ISPs. Since IoT devices are located far from cloud computing centres, a survey from the ICA Consortium reports that when the distance between IoT devices and cloud data centres is reduced by 322 km, the spending on data processing is reduced by 30% [21]. Figure 4 shows the data cost reduction schematic provided by the ICA Consortium, where the cost of data processing will gradually decrease as the computing distance gradually moves from the remote data center closer to the data source side. When AI technology is empowered into edge computing, the processing cost of data will be further reduced. On the other hand, maintaining many cloud computing devices will generate a large amount of energy overhead, and how to design a green communication and computing network will also be an important challenge now.

The computational distance moves from the remote data center to the data source, and the cost of data processing gradually decreases, and here the results are not linear because the relationship between the processed data and the processor is not linear. To have an intelligent view in complex driving environments, it is necessary to complete the processing of a large amount of data in a relatively short period and sense the current real-time traffic conditions, target characteristics, and pedestrian density to achieve a smooth driving pattern and experience. However, limited by the limited resources of the vehicle itself and the uneven distribution of resources among vehicles, satisfactory performance indicators, including data throughput, service experience, reliability, coverage, and other performance, are

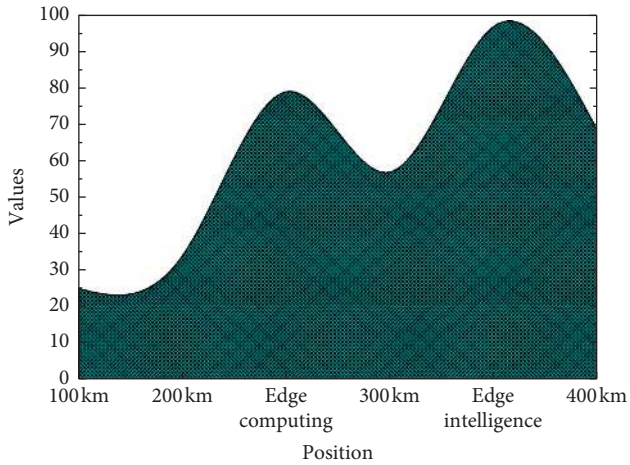

Figure 4: Diagram of data processing cost reduction (%) results.

usually not available, which poses a great challenge to the implementation and popularization of autonomous driving. On the other hand, our daily travel will generate a large amount of data, how to activate existing data, integrate local data, connect information islands into networks, establish data and information resources sharing mechanism, and other information service research is the urgent need to solve the problem of future intelligent transportation. At present, the combination of vehicle local computing and remote cloud computing platforms is the main computing mode to realize data processing and analysis. However, the limited computing capacity of vehicles and the unstable backhaul links between vehicles and cloud computing platforms will significantly increase the service latency of services, which will not satisfy the latency-sensitive telematics applications. Therefore, computational network system architecture and information service research strategies need to be designed for complex telematics applications to enhance the service level of smart transportation.

$$\nabla b(t) = E\{L(b(t+1)) + L(b(t))\}, \tag{9}$$

$$L(b(t+1)) - L(t) \geq \frac{1}{2} E\{L(b(t+1)) + L(b(t))\}, \tag{10}$$

where b is the corresponding flux, E is the total energy, L is the length, and t is the time. In complex IoT application scenarios, a common access technology does not exist that can meet the different distances and diverse network performance requirements. The future of wireless access networks will be a scenario of ultradense network deployment and the coexistence of multiple access technologies. Different types of IoT applications require matching connectivity to ensure service continuity [22–28]. However, the current network management architecture lacks the cooperation of convergence between different networks; for example, it is difficult to share information and resources between networks in real time. This relative independence of resources and networks does not provide a complete IoT solution. Besides, the widely distributed IoT devices generate a huge amount of multidimensional fragmented data, and the traditional cloud computing model cannot achieve real-time response for end-to-end services, while the existing edge access network does not play the natural advantage of being close to the data source end, but still only serves as a transmission pipeline for data, lacking the ability to sense and analyse the service traffic, device characteristics, and resource status, and thus cannot provide the upper control plane with refined resource management information.

4. Results and Discussion

Simulation results for the different number of users show that the proposed algorithm achieves lower system energy consumption than the other four algorithms. A negative energy consumption indicates that the system gains extra energy, thanks to the additional channels that provide more energy transmission; moreover, the superiority over the algorithms is because there is no limitation of cochannel interference, which provides more transmission power in the transmit energy channel, there is no attenuation factor of the power splitting ratio, so more energy is gained. Also, as the number of mobile devices increases, the system energy performance of several other algorithms starts to decrease, while this performance of the algorithm proposed in this chapter decreases insignificantly, also thanks to the availability of redundant channels to send energy without the influence of the transmit power limit and the attenuation of energy reception by the splitting ratio. Of course, the system energy performance still decreases as the number of users increases, which is due to more users, making the total equivalent distance of transmitted energy increase, as shown in Figure 5.

A scheme using additional channels to transmit energy is proposed, so this section compares the effect of the number of channels $M = N + C$, $C = 2, 4$, on the system performance. Other parameters are described as before, and the simulation results are shown in Figure 5. As a reference, the results of the algorithm on the energy consumption part of the system are put together with the results in Figure 5. From Figure 5, it is known that when the number of channels used for energy transmission is increased from 2 to 4, the system performance is further improved because more channels are available for energy transmission. And the trend of system energy consumption with an increasing number of users is the same. Therefore, increasing the number of channels will generally improve the system performance, but not linearly.

Figure 6 verifies the service latency of the service based on the two computational migration strategies for a different number of trials. Since VE-MACN is a distributed computational migration architecture, so the tasks of all vehicles adopt parallel transmission and computation modes, this paper adopts the cumulative average service latency of the services to reflect the service latency of all services of the VE-MACN computational migration system. As can be seen from the figure, the cumulative average service delay of the services of both computational migration strategies converges to a stable value as the number of trials increases, which indicates that the computational migration strategy based on deep augmentation learning has good convergence

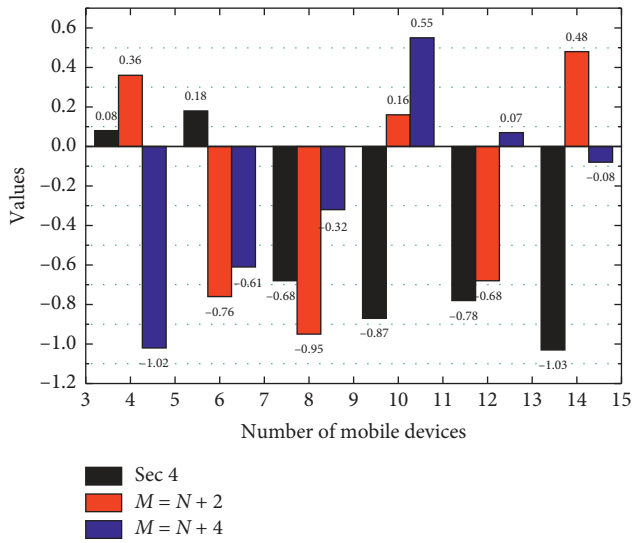

Figure 5: Comparison of the system performance with different numbers of channels.

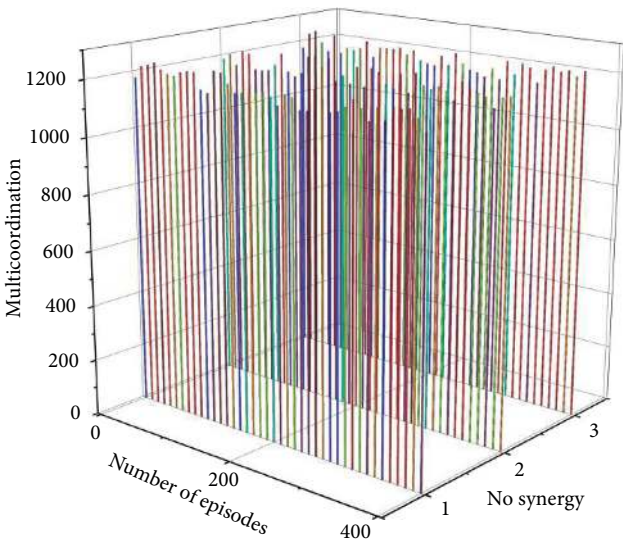

Figure 6: Comparison of average energy consumption of the VE-MACN system.

performance. On the other hand, compared with the computational migration strategy without vehicle collaboration, the multisite collaborative computational migration strategy proposed in this paper can reduce the average service delay of the service by 22.5%. The main reason is that when more vehicles with idle resources join the VE-MACN, vehicles with computational migration needs under the same traffic area use the multiaccess and task distribution strategy to migrate some tasks to roadside units and vehicles with idle resources to achieve parallel transmission and computation of data streams, thus reducing the service latency of the service.

We analyze the energy consumption comparison results under the two computing migration strategies. As can be seen from the figure, the cumulative average energy consumption values of the system under both computational migration strategies converge to an interval with an error of ±5 when the number of trials is greater than 200. In terms of the total system energy consumption, when there are no vehicles in VE-MACN to join the collaborative computational migration, the roadside unit needs to allocate more resources to vehicles with computational migration demand and then meet the service delay demand of users, which will increase the system energy consumption in this case. On the contrary, when vehicles with idle resources join the VE-MACN scenario, the multisite collaborative migration strategy proposed in this paper can be used to distribute tasks to the computational servers of both the roadside unit and the vehicles, and this distributed computing mode will reduce the number of tasks distributed for each edge node and also reduce the demand for computational resources of the edge nodes accordingly, thus reducing the overall energy consumption of the system.

When the weight parameter lies in the interval [0.1, 0.6], a smaller weight parameter implies that the demand for resources by task owners is higher. In Figure 7, when the number of task owners is small, differential pricing can satisfy the resource demand of task owners to the maximum extent, so the total satisfaction and average revenue of task owners are significantly improved. When the number of task owners is greater than 6, the number of resources allocated to each task owner gradually decreases because the current supply of resources is smaller than the demand for resources, so it will slow down the upward trend of total user satisfaction. On the contrary, the downward trend of the average revenue of task owners is obvious. This is because the current shortage of resource supply causes the price of resources to increase, thus reducing the average revenue of task owners. The weights located in the interval [1.0, 6.0] reflect the lower level of demand for resources by task owners compared to the smaller μ. The total satisfaction of task owners still shows an increasing trend even in the case of insufficient resource supply. Due to the gradual increase in the price of resources, the upward trend of their average returns will gradually smooth out when the number of task owners is high. The results in Figure 7 also demonstrate that task owners can change the matching of demand and resources by adjusting the weight parameter and can obtain greater overall satisfaction and average gain when the weights are larger.

Figure 8 compares the comparison results of the average service latency under different task offloading algorithms. Based on the distributed data transfer and parallel computing model, the average service delay under both task offloading strategies shows a decreasing trend as the number of computing resource providers increases. With the increase of the number of computing resource providers, the average service delay under the task offloading strategy proposed in this section is only slightly larger than that of the CTC-DP strategy. However, the complexity of the QPSO-based task offloading algorithm is much lower than that of ($|W_2||X_2||M|$) in the CTC-DP scheme. The complexity of the QPSO-based task offloading algorithm is (SP) for a given maximum number of iterations S and number of particles P. Moreover, the impact of the CTC-DP scheme on the

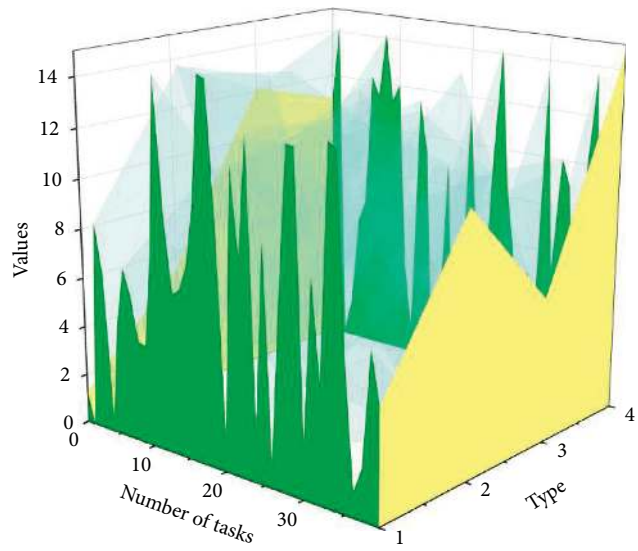

Figure 7: Total task owner satisfaction and average gain underweighting parameter $\mu = [0.1, 0.6]$.

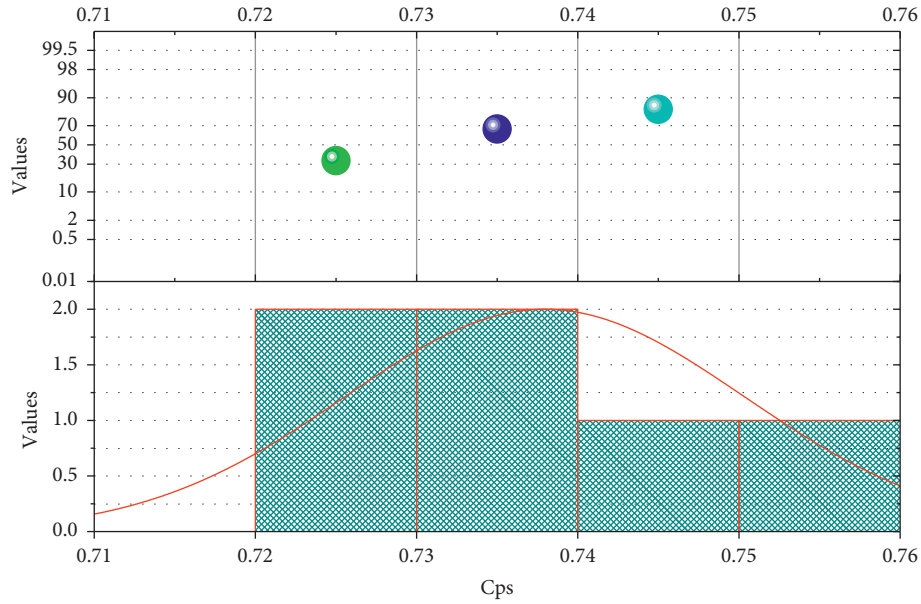

Figure 8: Comparison of average service latency with different task offloading algorithms.

performance depends not only on the dimensionality of the variables but also on the degree of discretization of the decision variables. In contrast, the task offloading strategy proposed in this section is not bounded by this rule. Therefore, an effective balance between business service latency and computational complexity is required to meet the stringent latency performance requirements.

Our model is highly efficient and has a high degree of accuracy. Unlike the traditional proof-of-work mechanism based on the "mining" process, this section designs a reputation evaluation mechanism that combines resource transactions and task offloading to provide a trusted computing environment and avoid taking up too many computing resources and generating large energy consumption. The user with the highest reputation value is responsible for packaging and writing the transaction records and reputation values into the blockchain. The system feasibility analysis and simulation results verify that the strategy proposed in this chapter can not only provide a trusted computing environment but also significantly reduce the signalling overhead and energy consumption of the blockchain system. By analysing this mixed-integer nonlinear programming model, the conclusion that its relaxed optimization problem is convex is obtained, and the Lagrange dual method is used to obtain the optimal unloading strategy, transmit power allocation, local computing power scheduling, and energy-carrying channel selection strategy, and the algorithm to minimize energy consumption is obtained. Then, based on this algorithm, a branch-and-bound approach is proposed to solve the algorithm for the

global optimal solution of the optimization problem. The final simulation comparison verifies the advantages of the algorithm proposed in this chapter in improving the energy consumption performance of the system.

5. Conclusion

In this thesis, a dynamic offloading and resource scheduling optimization model for a multiuser mobile edge cloud SWIPT system is developed through system computation and energy model analysis, and then this nonconvex optimization problem is transformed into a zero pairwise gap optimization problem. With the optimization algorithm of the pairwise problem, the results of this thesis make it possible to minimize the system energy consumption while satisfying the optimal policy requirements of clock frequency control, transmission power allocation, unloading ratio, and received power split ratio. Finally, the algorithm is verified by comparative simulations to show that the account has a good performance in terms of system energy consumption. By digitizing the distributed computing migration algorithm, the resource transactions and task offloads that comprise the reputation value evaluation will be embedded in the blockchain in the form of smart contracts to prevent malicious nodes from tampering with the transaction records and reputation values. Based on the Stackelberg model, joint differential pricing and joint optimization strategies for resource allocation are designed to achieve a balance of interests between task owners and resource providers. After completing the resource transaction, the smart contract automatically executes task offloading to achieve an effective compromise between business processing latency and algorithm complexity. After passing the validation of the computation results, the reputation value of each user is updated based on the resource allocation results and computation performance. The security, feasibility analysis, and numerical results show that the smart contract and consensus mechanism proposed in this paper can be effectively applied in a blockchain-enabled system.

Data Availability

Data sharing is not applicable to this article as no datasets were generated or analysed during the current study.

Consent

Informed consent was obtained from all individual participants included in the study references.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

References

[1] A. Shahidinejad and M. Ghobaei-Arani, "Joint computation offloading and resource provisioning for edge-cloud computing environment: a machine learning-based approach," *Software: Practice and Experience*, vol. 50, no. 12, pp. 2212–2230, 2020.

[2] C. Shu, Z. Zhao, Y. Han, G. Min, and H. Duan, "Multi-user offloading for edge computing networks: a dependency-aware and latency-optimal approach," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 1678–1689, 2019.

[3] W. Sun, J. Liu, and Y. Yue, "AI-enhanced offloading in edge computing: when machine learning meets industrial IoT," *IEEE Network*, vol. 33, no. 5, pp. 68–74, 2019.

[4] H. Guo, J. Liu, and J. Lv, "Toward intelligent task offloading at the edge," *IEEE Network*, vol. 34, no. 2, pp. 128–134, 2019.

[5] Z. Liao, J. Peng, B. Xiong, and J. Huang, "Adaptive offloading in mobile-edge computing for ultra-dense cellular networks based on genetic algorithm," *Journal of Cloud Computing*, vol. 10, no. 1, pp. 1–16, 2021.

[6] C.-H. Chu, "Task offloading based on deep learning for blockchain in mobile edge computing," *Wireless Networks*, vol. 27, no. 1, pp. 117–127, 2021.

[7] Z. Li, V. Chang, J. Ge, L. Pan, H. Hu, and B. Huang, "Energy-aware task offloading with deadline constraint in mobile edge computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2021, no. 1, pp. 1–24, 2021.

[8] A. Yousafzai, I. Yaqoob, M. Imran, A. Gani, and R. Md Noor, "Process migration-based computational offloading framework for IoT-supported mobile edge/cloud computing," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4171–4182, 2019.

[9] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT security techniques based on machine learning: how do IoT devices use AI to enhance security?" *IEEE Signal Processing Magazine*, vol. 35, no. 5, pp. 41–49, 2018.

[10] P. Cong, J. Zhou, L. Li, K. Cao, T. Wei, and K. Li, "A survey of hierarchical energy optimization for mobile edge computing: a perspective from end devices to the cloud," *ACM Computing Surveys (CSUR)*, vol. 53, no. 2, pp. 1–44, 2020.

[11] D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne, "Privacy-preserved task offloading in mobile blockchain with deep reinforcement learning," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2536–2549, 2020.

[12] L. Hu, Y. Tian, J. Yang, T. Taleb, L. Xiang, and Y. Hao, "Ready player one: UAV-clustering-based multi-task offloading for vehicular VR/AR gaming," *IEEE Network*, vol. 33, no. 3, pp. 42–48, 2019.

[13] S. Shen, Y. Han, X. Wang, and Y. Wang, "Computation offloading with multiple agents in edge-computing–supported IoT," *ACM Transactions on Sensor Networks (TOSN)*, vol. 16, no. 1, pp. 1–27, 2019.

[14] J. Cao, D. Zhang, H. Zhou, and P.-J. Wan, "Guest editorial emerging computing offloading for IoTs: architectures, technologies, and applications," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 3987–3993, 2019.

[15] Y. Chen, Y. Zhang, Y. Wu, L. Qi, X. Chen, and X. Shen, "Joint task scheduling and energy management for heterogeneous mobile edge computing with hybrid energy supply," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8419–8429, 2020.

[16] B. Yang, D. Wu, and R. Wang, "CUE: an intelligent edge computing framework," *IEEE Network*, vol. 33, no. 3, pp. 18–25, 2019.

[17] J. Du, *Research on Multidimensional Resource Management Strategies in Mobile Edge Computing Networks*, Xi'an University of Electronic Science and Technology, Xi'an, China, 2018.

[18] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: a comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 869–904, 2020.

[19] L. Yang, X. Chen, S. M. Perlaza, and J. Zhang, "Special issue on artificial-intelligence-powered edge computing for Internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9224–9226, 2020.

[20] H. Hu, H. Shan, C. Wang et al., "Video surveillance on mobile edge networks-A reinforcement-learning-based approach," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4746–4760, 2020.

[21] W. Huang, K. Ota, M. Dong, T. Wang, S. Zhang, and J. Zhang, "Result return aware offloading scheme in vehicular edge networks for IoT," *Computer Communications*, vol. 164, pp. 201–214, 2020.

[22] M. Cui, S. Zhong, B. Li, X. Chen, and K. Huang, "Offloading autonomous driving services via edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10535–10547, 2020.

[23] J. Shi, Y. Lu, and J. Zhang, "Approximation attacks on strong PUFs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 2138–2151, 2019.

[24] Z. Zhang, M. Liu, M. Zhou, and J. Chen, "Dynamic reliability analysis of nonlinear structures using a duffing-system-based equivalent nonlinear system method," *International Journal of Approximate Reasoning*, vol. 126, pp. 84–97, 2020.

[25] W. Wei, Q. Ke, J. Nowak, M. Korytkowski, R. Scherer, and M. Woźniak, "Accurate and fast URL phishing detector: a convolutional neural network approach," *Computer Networks*, vol. 178, Article ID 107275, 2020.

[26] S. Mehta and P. Kaur, "Efficient computation offloading in mobile cloud computing with nature-inspired algorithms," *International Journal of Computational Intelligence and Applications*, vol. 18, no. 4, Article ID 1950023, 2019.

[27] K. Sim, J. Yang, W. Lu, and X. Gao, "Blind stereoscopic image quality evaluator based on binocular semantic and quality channels," *IEEE Transactions on Multimedia*, p. 1, 2021.

[28] W. Wang, N. Kumar, J. Chen et al., "Realizing the potential of the Internet of things for smart tourism with 5G and AI," *IEEE Network*, vol. 34, no. 6, pp. 295–301, 2020.