mathematics

MDPI

Article

Risk Assessment Edge Contract for Efficient Resource Allocation

Minghui Sheng [1], Hui Wang [1,*], Maode Ma [2], Yiying Sun [1] and Run Zhou [3]

[1] College of Computer Science and Technology, Zhejiang Normal University, Jinhua 341000, China
[2] College of Engineering, Qatar University, Doha 974, Qatar
[3] College of Mechanical and Electrical Information, Yiwu Industrial and Commercial College, Jinhua 322000, China
* Correspondence: hwang@zjnu.cn

Abstract: The rapid growth of edge devices and mobile applications has driven the adoption of edge computing to handle computing tasks closer to end-users. However, the heterogeneity of edge devices and their limited computing resources raise challenges in the efficient allocation of computing resources to complete services with different characteristics and preferences. In this paper, we delve into an edge scenario comprising multiple Edge Computing Servers (ECSs), multiple Device-to-Device (D2D) Edge Nodes (ENs), and multiple edge devices. In order to address the resource allocation challenge among ECSs, ENs, and edge devices in high-workload environments, as well as the pricing of edge resources within the resource market framework, we propose a Risk Assessment Contract Algorithm (RACA) based on risk assessment theory. The RACA enables ECSs to assess risks associated with local users by estimating their future revenue potential and updating the contract autonomously at present and in the future. ENs acquire additional resources from ECSs to efficiently complete local users' tasks. Simultaneously, ENs can also negotiate reasonable resource requests and pricing with ECSs by a Stackelberg game algorithm. Furthermore, we prove the unique existence of Nash equilibrium in the established game, implying that equilibrium solutions can stably converge through computational methods in heterogeneous environments. Finally, through simulation experiments on the dataset, we demonstrate that risk assessment can better enhance the overall profit capability of the system. Moreover, through multiple experiments, we showcase the stability of the contract's autonomous update capability. The RACA exhibits better utility in terms of system profit capabilities, stability in high-workload environments, and energy consumption. This work provides a more dynamic and effective solution to the resource allocation problem in edge systems under high-workload environments.

Keywords: mobile edge computing; resource allocation; resource pricing; risk assessment; Stackelberg game; Nash equilibrium

MSC: 91-10

Citation: Sheng, M.; Wang, H.; Ma, M.; Sun, Y.; Zhou, R. Risk Assessment Edge Contract for Efficient Resource Allocation. *Mathematics* **2024**, *12*, 983. https://doi.org/10.3390/math12070983

Academic Editor: Hsien-Chung Wu

Received: 5 March 2024
Revised: 16 March 2024
Accepted: 23 March 2024
Published: 26 March 2024

1. Introduction

Due to the rapid growth of edge devices and mobile applications, wireless networks have been facing unprecedented data traffic pressures in recent years. Consequently, edge computing has emerged as a new paradigm to complement cloud computing, bringing computing capabilities closer to end-users to meet the demands of fast and flexible computing and communication-enhanced services [1,2]. Network operators or cloud service providers can utilize Edge Computing Servers (ECSs) deployed at base stations and edge devices as edge nodes (ENs) by Device-to-Device (D2D) communication to assist in offloading. The edge nodes, which could be idle mobile devices, may exhibit high heterogeneity in computing capabilities and costs, providing complementary services. It provides edge computing with many significant functionalities, including local data processing and analysis, distributed caching, localization, resource pooling, scaling, enhanced privacy and security,

and reliable connectivity [3,4]. Therefore, edge computing is crucial for meeting stringent requirements of emerging systems and low-latency applications such as embedded artificial intelligence, virtual/augmented reality (VR/AR), and the future Internet.

However, edge computing also faces numerous challenges that need to be addressed. For instance, in the context of regional services, resource pooling composed of ECSs and multiple ENs introduces challenges of edge resource allocation and resource sharing among edge devices. ECSs have tasks to provide scheduling services for computing resources, while ENs, which are assumed to be idle edge devices, primarily focus on executing the computational tasks. Unlike the capacity of traditional clouds, the computational capability of regional resource-sharing centers, ECSs, is limited. Additionally, around an ECS, there may be numerous distributed ENs with different capacities and configurations. Due to the heterogeneity of these ENs in terms of location, specifications, reliability, and reputation, the ECS services may exhibit different preferences towards them. For example, in the context of edge home robots, the accuracy of tasks like object manipulation possesses higher significance, while a slight compromise in processing speed is acceptable. Conversely, in the realm of connected vehicles, data transmission speed often outweighs the precision of location data, enabling the system to swiftly respond to traffic conditions for effective traffic management.

Therefore, considering service priorities and fairness, the primary focus is on effectively allocating limited edge resources to complete services with different characteristics and preferences. The challenges we face can be summarized as follows:

1. In a resource pool composed of multiple ECSs and ENs, when an EN receives requests from edge devices, it sometimes needs to request computing resources from an ECS to complete the tasks according to its computing capabilities and the demands of edge devices. It is important to learn how to determine whether an EN should request computing resources from the ECS and how to determine the appropriate amount of requested resources to maximize the service quality for the edge devices in the region.
2. Edge devices are often different with heterogeneity in terms of location, specifications, reliability, and the reputation of edge devices and/or ENs. The allocation of computing resources is profoundly influenced. For instance, from a reputation perspective, within a resource pool, certain edge devices and ENs might frequently engage in resource monopolization, excessively occupying computing resources or requesting computing resources from an ECS at a high frequency. In such cases, the ECS tends to allocate resources to the edge devices with lower risk profiles or might raise the resource prices for high-demand ENs, while the ECS itself possesses specific resource preferences. For instance, in an edge hospital setting, online clinics often prioritize image data and tend to allocate more computing resources to image-related tasks compared to audio data. The heterogeneity of computational demands and the preferences of the ECS constitute significant challenges in edge resource allocation.
3. Since the usage of user data is increasing day by day, the data usage of users can be enormous. In some environments that are highly sensitive to computational latency and have strong requirements for real-time information, known as high-workload resource scheduling environments, the computing resources of an ECS could be fully occupied for extended periods. In such cases, the scheduling algorithms often lead to issues like energy waste and scheduling delays. Designing more efficient allocation schemes to reduce energy waste, scheduling delays, and edge service failure rates in high-workload resource scheduling environments is crucial.
4. The computing resource allocation for edge devices needs to consider a joint optimization model of resource allocation and task offloading, which often involves NP-hard optimization problems. If the algorithms cannot meet the required time complexity for optimization problems, facing an exponential growth trend in data volume, significant delays may occur. Therefore, the algorithms must be capable of fast convergence.

In this paper, we propose a resource trading mechanism based on Stackelberg game and risk assessment contracts. This approach leverages the principles of market economy

theory, where sellers assess the risk of buyers and formulate transaction contracts, to address the challenges of resource pricing and scheduling within a resource pool consisting of multiple ECSs and ENs in the edge system. In the considered model, a portion of edge devices act as ENs and assist in offloading through D2D connections. The ECSs and ENs have different levels of computing capabilities, and the edge devices within the resource pool region issue demand the subscribed ECS or ENs in their vicinity. The ENs have their own computing capabilities and, to better serve the edge devices, they analyze network traffic and resource utilities before purchasing a portion of computing resources from the ECS, which, as a resource operator or provider, determines how to coordinate the requested tasks from the ENs to match its computing resources. Following the Stackelberg game model, the ECS is considered the leader in the resource market, while the ENs are considered followers. The edge devices, as the initiators of computing tasks, become local task demanders once they establish subscription relationships. Furthermore, the resource allocation issue resembles an investment behavior in financial markets, where the preference of the ECS on edge resource allocation signifies the preference of investors or venture capital for investment in the financial market.

In this paper, we combine the heterogeneity of edge devices with risk assessment theory in investments. By enabling the ECS and ENs to sign edge contracts, the ECS exhibits superior performance in dealing with resource allocation preferences and the heterogeneity of edge devices, making the market model more realistic to provide new solutions for the edge resource allocation and task offloading problems. Finally, using the Lagrange multiplier method, we transform the joint optimization problem of multi-to-multi resource allocation and task offloading based on risk assessment edge contracts into a convex optimization problem. We design a heuristic algorithm with a quick convergence to solve the optimization problem.

The major contributions in this paper are as follows:

1. We develop a resource trading model based on the Stackelberg game for solving task offloading problems in a resource pool composed of ECSs and ENs. The model allocates computing resources according to the proportion of EN requests determined by the game, while the existence of Nash equilibrium in the game is proved by using fixed-point theory.
2. By incorporating risk assessment theory, we introduce the concept of edge risk to evaluate the profits and benefits based on the contracts with edge devices. The data usage follows a generalized Wiener process, and the ECS predicts the future edge risk value that edge devices will possess based on their current data volume. It provides a unified standard for heterogeneous edge devices. A novel edge contract based on the edge risk assessment is proposed, aiming to reduce transaction delays and energy losses in high-workload environments for resource allocation and task offloading. The contract allows edge devices to autonomously update two contract elements: the limit of local computation resource request and the computation resource price, in real time.
3. To address the resource allocation problem between the ECSs and the ENs, we design a low-complexity risk assessment contract algorithm (RACA). Furthermore, we prove the strict convexity of the subgame problem and solve it using the Lagrange multiplier method to find the equilibrium point in the game model. The heuristic algorithm demonstrates the capability of swift convergence, guided by the proportion of user-requested resources to the overall requested resources, allocating the computing resources acquired by the ENs to their respective edge devices.
4. We conduct simulation experiments to validate the convergence efficiency of the designed algorithm under high-workload conditions. Furthermore, we evaluate the performance of the ECS, utilizing the proposed RACA algorithm, which demonstrates its superiority compared to other similar algorithms. Additionally, the simulation results show the advantages of the proposed request ratio allocation scheme when the

ENs receive specific requested resources. The performance of the RACA algorithm in terms of game convergence is demonstrated, too.

The remaining parts of this paper are summarized as follows: In Section 2, a review of related works is presented. Section 3 describes the edge resource pool model composed of multiple ECSs, ENs, and edge devices in a high-workload environment and introduces the edge contract framework based on risk assessment. In Section 4, specific problems are formulated, and the convexity of the subgame problem is verified. The Lagrange multiplier method is introduced, and the equilibrium points of the game between the ECS and ENs are derived using the backward induction method. Section 5 presents and discusses a heuristic algorithm called RACA based on risk assessment edge contracts and explores the game algorithm for finding the Nash equilibrium point between the ECS and ENs. In Section 6, simulation experiments to evaluate the performance of the RACA algorithm and the game allocation algorithm are presented. Finally, in Section 7, a comprehensive summary of the proposed framework and algorithms is provided along with a discussion of potential future improvements.

2. Related Work

Researchers have explored various aspects to effectively allocate limited edge resources to computing services with different characteristics and preferences. Bozkaya [5] proposed a digital twin framework to simulate real-world edge environments based on minimizing processing time and access latency costs. Luo [6] introduced a small-scale edge caching scheme by removing duplicate data through an edge caching system. However, these works focus on optimizing edge devices or content delivery without considering the interactions among the participants in an edge ecosystem.

To address the competition among edge devices with different characteristics and preferences, market theory has been applied to simulate the differences among edge devices. By modeling the market competition, edge devices can make better resource allocation decisions to maximize social welfare. Market pricing mechanisms have been widely used to solve edge resource allocation problems, directing the allocations of limited resources that maximize social benefits. They enable edge devices to autonomously choose resource allocation and scheduling strategies to cope with the overwhelming information volume. Yuan [7] considered a decentralized blockchain edge solution based on a market framework, but it lacked comprehensive consideration for computing latency and energy consumption. Additionally, some researchers have devised several resource allocation mechanisms to ensure user QoS, such as convex optimization-based approaches [8] and coordination through composite tables [9]. Additionally, some joint optimization models have been developed for task offloading and resource allocation [10] with several scheduling algorithms including deep Q-learning [11,12] and ADMM [13]. However, these joint optimization models work based on nonlinear programming and are typically NP-hard, being difficult to find optimal solutions in polynomial time [14,15].

On the other hand, for edge devices participating in resource allocation in a market environment, challenges arise in terms of the leakage of privacy information and unfair resource allocation. Some allocation methods based on edge risk have been proposed. Rui Chen, Ganesh Neelakanta Iyer [16,17], and others considered the interactions among market participants with different risk attitudes and incorporated risk-neutral computation with given weights. Kai Peng [18] studied the privacy risk of different edge devices to propose a privacy-preserving game without considering the impact of risk on the allocation. Jianbo Du [19] used the asynchronous advantage actor-critic (A3C) deep reinforcement learning algorithm to obtain resource pricing and allocation and explored the risk coefficients that may exist using reinforcement learning. However, the required computational complexity remains significant for more lightweight edge devices. Minghui Liwang [20] introduced the concept of financial futures in unmanned aerial vehicles and vehicular networks and designed a mutually beneficial and risk-tolerant forward contract to facilitate pricing and gaming. Vladimir Marbukh [21] replaced user rate/throughput with risk entropy rate (ERaR)

and incorporated risk pricing into the overall system optimization for users with different risk tolerance levels. However, issues such as a lack of generality and inefficiency still persist.

A comparison of related work is presented in Table 1. In conclusion, our proposed approach draws on valuable insights from existing research on edge resource allocation in the academic community. Our solution aims to address some of the challenges present in the current studies, particularly in high-workload edge environments.

Table 1. Comparison of related works.

Paper	Edge-Cloud	D2D	Economic	Delay	Energy	Prediction	High Workload
[5,6,10]	✓			✓	✓		
[7]		✓	✓			✓	✓
[8,9]	✓			✓			✓
[11]				✓		✓	
[12]		✓				✓	
[13]		✓		✓	✓		✓
[14]		✓	✓	✓			
[15]	✓			✓	✓		✓
[16]	✓		✓	✓			
[17]	✓		✓	✓		✓	
[18]		✓	✓	✓	✓		
[19]	✓	✓	✓			✓	
[20]	✓		✓	✓	✓	✓	
[21]	✓		✓	✓		✓	
Ours		✓	✓	✓	✓	✓	✓

3. System Model

In this section, we first describe the operational mechanism of contracts, followed by a discussion on the changes in the data from edge devices and the profit models for ECSs and ENs, drawing reference from the D2D model proposed by C. Yi and P. Dai [22,23]. Finally, we present the problem of risk assessment.

Initially, we consider a high-workload area with multiple edge devices as users (User), denoted as $I = \{1, 2, \ldots, I\}$, where i represents the index of I . User requests are sent to the surrounding ENs and the corresponding ECS. $J = \{1, 2, \ldots, J\}$ is used to represent ECSs and $K = \{1, 2, \ldots, K\}$ is to represent the ENs, where j and k are the indices of J and K, respectively. The notations and descriptions are shown in Table 2. Due to the limited availability of resources at an EN, when an EN has received tasks beyond its acceptable resource threshold or existing resources, it is insufficient to meet real-time user requests. The EN has to purchase additional computing resources from the nearby ECS to cater to the needs of subscribed users. As illustrated in Figure 1, in the resource pool, multiple ECSs (Edge Computing Servers) and multiple users communicate with each other, with users communicating via D2D (Device-to-Device) communication. Users with more (or idle) computing resources may act as ENs (Edge Nodes) to assist in task offloading. ECSs and ENs have different levels of computing capabilities, and users within the resource pool area send demands to subscribed ECSs or ENs in the vicinity. ENs have their own computing capabilities, and to better serve users, they combine channel networking and resource utility analysis to purchase a portion of computing resources from ECSs. The ECS serves as a seller of computing resources, while the ENs act as buyers of resources. The ECS handles requests from the ENs while simultaneously handling the requests from the local users subscribed to the ECS. Each user can send a request to either an ECS or an EN, and each EN purchases a specific amount of computing resources from its nearby ECS.

3.1. Problem Overview and Contract Model

In low-workload scenarios, the computing resources of an ECS are able to meet the computational demands of the entire resource pool. However, in high-workload environments, the local requests received by the ECS along with the purchasing requests from ENs can surpass its total capacity of computing resources. To efficiently utilize all computational resources, the ECS needs to strike a balance between the local users and the ENs while minimizing energy consumption and latency due to on-site transactions in high-workload conditions. Additionally, the ECS should exhibit self-adaptability and self-optimization to handle various situations including prolonged high-demand periods, sudden bursts of demand, and the presence of malicious user requests in the system.

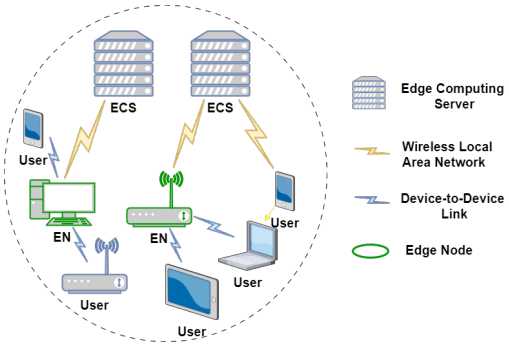


Figure 1. Regional resource pool model with multiple ECSs, ENs, and users.

To address these challenges, we design an edge contract mechanism based on risk assessment. As shown in Figure 2, the ECS has a large computing resource capacity denoted by Q. Upon receiving local user resource requests (fl) and computing resource requests from the ENs (fen), the ECS makes allocations based on its own computing resources (Q) and the pre-defined local resource allocation limit (L) specified in the contract. The actual allocated resources for local users and ENs are represented by Fl (shown in blue) and Fen (shown in white), respectively, with Cp indicating the amount of resources to be compensated to local users. To maximize the utilization of the resources, the sum of the allocated resources for local Fl and EN Fen always equals Q in each allocation, ensuring that all computing resources are allocated.

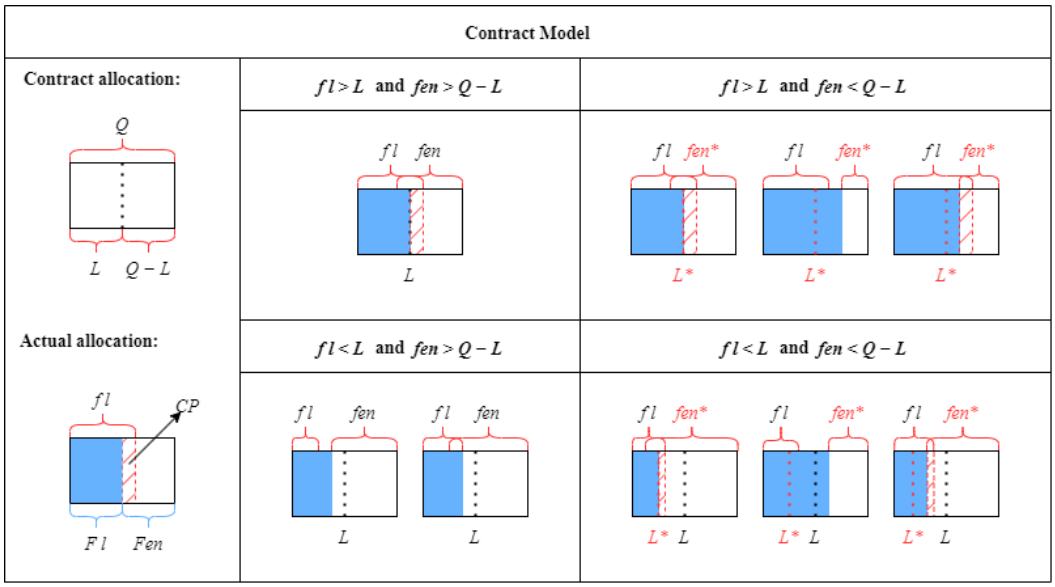


Figure 2. Edge contracts based on risk assessment of contract value L and several examples.

The contract, denoted as (L, p), consists of two components including the pre-defined local request limit (L) and the resource contract price for the ENs (p). The ECS utilizes this contract to handle different scenarios, efficiently avoiding on-site transaction drawbacks.

In the first scenario, when $fl > L$ and $fen > Q - L$, it signifies that the total user demand surpasses the contract limit, and the ENs are also inclined to acquire a substantial amount of computing resources. To expedite transactions and mitigate the wastage of resources and the challenges arising from mutual bargaining, we employ the contract mechanism. Guided by the specified local resource upper limit (L) in the contract, we allocate LL resources to the local users and $Q - L$ resources to the ENs. This allocation strategy ensures efficient and swift trading to prevent the drawbacks of resource wastage and negotiation challenges among the parties involved.

In the second scenario, when $fl > L$ and $fen < Q - L$, it implies that due to the contract limits, the ECS has only allocated its resources to local users, yet ENs do not have as much demand at the contract price (p). In such cases, the contract becomes less effective due to the mismatch between user demands and the allocated resources. In this case, the ECS updates the contract (L^*, p^*) based on the risk measurement, where the new local resource limit $L^* < fl$. The ENs evaluate the new contract price p^* and generate a new computing resource demand fen^*. If $fen^* > Q - L^*$, similar to the first scenario, the ECS allocates fen^* resources to the ENs and the remaining to local users, ensuring that the ENs obtain the desired computing resources. Meanwhile, local allocation remains the same based on the original contract.

In the third scenario, when $fl < L$ and $fen > Q - L$, the situation arises where the demand for computing resources from local users (fl) does not exceed the contract's upper limit (L), but the ENs desire a substantial amount of computing resources. It is prioritized to satisfy the local users' needs. If the requests from the ENs have not reached the upper limit $Q - fl$, on-site transactions can be conducted based on the contract avoiding additional consumption. Ultimately, the local users receive $max(fl, Q - fen)$ resources, while the ENs are allocated $min(Q - fl, fen)$ resources.

In the fourth scenario, when $fl < L$ and $fen < Q - L$, both local users and ENs are requesting only a small amount of the computing resources resembling a low-workload environment. Generally, this situation is rarely encountered in high-workload systems. In this situation, the contract is ineffective, and the ECS needs to re-establish a new contract similar to the second scenario. By determining risk measurements, a new contract limit L^* ($L^* < fl$) can be obtained, and a new contract and resource allocation can be formulated as in the second scenario.

In conclusion, with the impact of the contract mechanism (L, p) or (L^*, p^*) in each scenario, the ECS allocates $Fl = max(min(L, fl), Q - fen)$ resources to local requests, and the ENs can purchase $Fen = min(max(Q - L, Q - fl), fen)$ resources. Based on this contract mechanism, the RACA algorithm is developed, which will be further detailed in next section.

Table 2. Notations and descriptions

Notations	Descriptions
I, J, K	Sets of users, ECSs, and ENs
i, j, k	Indices for users, ECSs, and ENs
$local_{i,j}$	Subscription of $User_i$ to ECS_j
$local_{i,k}$	Subscription of $User_i$ to EN_k
$sub_{j,k}$	Subscription of EN_k to ECS_j
S_i	Data generated by $User_i$
u_i, σ_i	Wiener process drift and volatility of $User_i$
α_i	Task request rate of $User_i$
Dl_i	Task request amount of $User_i$
$Dl_{i,j}$	Task request amount from $User_i$ to ECS_j
λl_i	Expected task completion time of $User_i$
pl_j	Unit gain for handling local tasks by ECS_j
p_j	Unit price of computing resources
cp_j	Unit price of compensatory resources
fl_j, fen_j	Local/EN computing resource requests to ECS_j

Table 2. *Cont.*

Notations	Descriptions
Fl_j, Fen_j	Final resource allocation by ECS_j to local/EN
UE_j, UN_k	Final revenue of ECS_j/EN_k
$Dtotal_k$	Local task volume of EN_k
ql_k	Computing resources held by EN_k
pl_k	Unit gain after EN_k saves time
$W_{j,k}$	Channel bandwidth between EN_k and ECS_j
$P_{j,k}$	Transmission power between EN_k and ECS_j
$H_{j,k}$	Channel gain between EN_k and ECS_j
$\omega_{j,k}$	Loss weight between EN_k and ECS_j
$E_{j,k}$	Energy loss between EN_k and ECS_j
$Uu_{i,j}$	Revenue contribution of $User_i$ to ECS_j
λu_j	Threshold for acceptable contribution increase
$R_{i,j}$	Risk of $User_i$ for ECS_j
λr_j	Threshold for acceptable risk for ECS_j

3.2. User Model

The unpredictable nature of the task requests from edge devices received by the ECS at different times poses significant challenges for the ECS to devise contracts and promote future transactions. In this subsection, we start with a user model to predict the quantities of future user task requests.

We assume that the data generated by each user, denoted by S, follow a log-normal distribution and the variation in the user exhibits Markovian properties. In a relatively stable system, the variation in data quantities follows a generalized Wiener process. Hence, the rate of change of data quantity with time can be expressed as follows:

$$dS = \mu \cdot S \cdot dt + \sigma \cdot S \cdot dz, dz = \epsilon\sqrt{dt} \tag{1}$$

where $\epsilon \sim N(0,1)$ represents the standard Wiener process and μ and σ are the drift and volatility coefficients of the Wiener process, respectively.

Considering Dl as the computational workload generated by a user which will request computational resources from either the ECS or an EN depending on the specific physical location of the user, we have:

$$Dl = \alpha \ln S \tag{2}$$

where $\alpha \sim U(A,B)$ represents the task request rate, following a uniform distribution, indicating the ratio of data generation to task requests during system operation. Let S^T be the data quantity at time T. Then, the computational workload generated by the tasks at time T, denoted as Dl^T, is given by:

$$Dl^T = \alpha \ln S^T \tag{3}$$

where Dl^0 represents the initial computational workload and S^0 represents the initial data quantity at the user. They are related as follows:

$$Dl^0 = \alpha \ln S^0 \tag{4}$$

According to the theory of Ito processes [24], we can obtain the expression $d(Dl) = \alpha\left(u - \frac{1}{2}\sigma^2\right)dt + \alpha\sigma dz$, which still satisfies the properties of a Wiener process. For the detailed derivation process, please refer to the Appendix A. We can express (3) as:

$$\begin{aligned} Dl^T &= Dl^0 + d(Dl) \\ &= \alpha \ln S^0 + d(\alpha \ln S) \\ &= \alpha \ln S^0 + \alpha\left(u - \frac{1}{2}\sigma^2\right)dt + \alpha\sigma dz \end{aligned} \tag{5}$$

Furthermore, we can obtain the mean and variance of the quantity Dl^T as follows:

$$E\left(Dl^T\right) = \alpha \ln S^0 + \left(u - \frac{1}{2}\sigma^2\right)\alpha T \tag{6}$$

$$Var\left(Dl^T\right) = \alpha^2\sigma^2 T \tag{7}$$

where $E(Dl^T)$ represents the mean of Dl^T and $Var(Dl^T)$ represents its variance. The ECS can utilize this process to predict the local workload variation and assess the computational efficiency risks of each user, which will be further elaborated upon in the risk assessment of contracts.

3.3. ECS Revenue Model

For $\forall j \in J$, the local computational resource requests fl_j, received by ECS_j, can be represented as:

$$fl_j = \sum_{i=1}^{I} \frac{Dl_{i,j}}{\lambda l_i} local_{i,j} \tag{8}$$

where λl_j represents the expected task completion time of $User_i$. If the completion time exceeds this limit, the task efficiency will decrease. $Dl_{i,j}$ represents the computational request quantity from $User_i$ received by ECS_j. $local_{i,j} \in \{0,1\}$ is a binary variable, where $local_{i,j} = 1$ indicates that $User_i$ is a local user subscribing to ECS_j. The final amount of computational resources allocated by ECS_j to local users, Fl_j, is calculated as $Fl_j = max(min(fl_j, L), Q - fen_j)$. The fen_j represents the total computational resources that ENs need to purchase from ECS_j.

Let pl_j represent the unit benefit that ECS_j gains from processing tasks for $User_i$. Then, the local revenue Ul_j for ECS_j is given by:

$$Ul_j = Fl_j pl_j \tag{9}$$

Now, let us consider all EN_k nodes denoted by $K = \{1, 2, \ldots, K\}$. The total computational resources fen_j requested by ECS_j from ENs can be represented as:

$$fen_j = \sum_{k=1}^{K} f_{j,k} sub_{j,k} \tag{10}$$

where $f_{j,k}$ represents the amount of computational resources EN_k requests from ECS_j and $sub_{j,k} \in \{0,1\}$ is a binary variable, where $sub_{j,k} = 1$ indicates that ECS_j is a subscribing node for EN_k.

We assume p_j is the unified price set by ECS_j for the computational resources allocated to each EN_k node. Then, the revenue Us_j received by ECS_j from ENs is given by:

$$Us_j = Fen_j p_j \tag{11}$$

where Fen_j represents the actual total amount of computational resources purchased by ENs, and its specific allocation strategy is determined by the edge contract, which satisfies $Fen_j = min(max(Q - fl_j, Q - L), fen_j)$.

By this model, the pre-signed resource contracts are adaptive to reduce connection failures, transaction failures, inadequate information timeliness, and high communication overhead during communication. The contract specifies the local resource upper limit L for each transaction. If the local request fl_j exceeds the upper limit L, ECS_j is responsible for completing the part that should be performed locally. However, if the computational task is not completed or delayed, ECS_j needs to compensate the local users. Considering that after the final allocation, if there are remaining computational resources, i.e., $Q - fen_j > L$, the

final computational resources allocated to the local users Fl_j are given by $Fl_j = Q - Fen_j$. Then, the compensation Cl_j provided by ECS_j to its local subscribers can be calculated as:

$$Cl_j = max\Big(fl_j - Fl_j, 0\Big)cp_j \tag{12}$$

where cp_j represents the unit price that ECS_j compensates to local users for each unit of computational resources. The compensation unit price cp_j should not exceed the local request benefit pl_j. If the actual computational resources allocated to the local users Fl_j are greater than their requested amount fl_j, then ECS_j does not need to compensate its local users, i.e., $Cl_j = 0$.

Therefore, the final revenue UE_j of ECS_j is given by:

$$UE_j = Ul_j + Us_j - Cl_j \tag{13}$$

3.4. EN Revenue Model

For each EN_k, $\forall k \in K$, the total local request received from $User_i$ is represented as $Dtotal_k$, and it is given by:

$$Dtotal_k = \sum_{i=1}^{I} Dl_{i,k} local_{i,k} \tag{14}$$

where $Dl_{i,k}$ represents the local request received by EN_k from $User_i$ and $local_{i,k} \in (0, 1)$ is a binary variable, where $local_{i,k} = 1$ indicates that $User_i$ is a local user subscribing to EN_k.

Let pl_k represent the unit benefit that EN_k gains by using computational resources purchased from ECS_j compared to processing users' tasks using its own computational resources. The unit benefit pl_k should be higher than the unit price $p_{j,k}$, at which EN_k purchases computational resources from ECS_j, i.e., $pl_k \geq p_{j,k}$. Then, the computation revenue Ub_k for EN_k is given by:

$$Ub_k = (\frac{Dtotal_k}{ql_k} - \frac{Dtotal_k}{ql_k + \sum_{j=1}^{J} f_{j,k} sub_{j,k}})pl_k \tag{15}$$

where ql_k represents the computational resources that EN_k possesses and ql_k should be less than the computational resources of ECS_j, i.e., $ql_k < Q_j$. $f_{j,k}$ represents the amount of computational resources EN_k requests from ECS_j. By the model, EN_k can only make one request to an ECS_j in the same round, i.e., $\sum_{j=1}^{J} sub_{j,k} \leq 1$. If EN_k can complete user tasks without requesting assistance, then EN_k does not send a request. In other words, when $\sum_{i=1}^{I} \frac{Dl_{i,k}}{\lambda l_i} local_{i,k} \leq ql_k, f_{j,k} = 0$.

The cost Cs_k for EN_k to subscribe computational resources from ECS_j is given by:

$$Cs_k = \sum_{j=1}^{J} f_{j,k} p_j sub_{j,k} \tag{16}$$

When EN_k and ECS_j conduct resource transactions, they will incur additional delay consumption and energy consumption. The delay consumption Ac_k for EN_k is given by:

$$Ac_k = \sum_{j=1}^{J} \left(\frac{f_{j,k}}{W_{j,k} \log\Big(1 + P_{j,k} H_{j,k}\Big)} + E_{j,k} \right) sub_{j,k} \omega_{j,k} \tag{17}$$

where $W_{j,k}$ represents the communication bandwidth. $P_{j,k}$ represents the transmission power. $H_{j,k}$ represents the channel gain, which includes fast fading gain and slow fading gain. $E_{j,k}$ represents additional fixed losses, and $\omega_{j,k}$ represents the weight of the loss.

Therefore, the final revenue UN_k for EN_k is given by:

$$UN_k = Ub_k - Cs_k - Ac_k \tag{18}$$

3.5. Contract Risk Assessment

In the proposed system, risks primarily arise from the uncertainty associated with the total local request, denoted as fl_j, and the prices of computing resources p_j, determined by the game between ECSs and ENs and the uncertainty of the resource purchase quantity of ENs, represented as $f_{j,k}$ Since the ECS, as the contract designer, cannot predict the resource purchase quantity of ENs, the prices p_j and resource purchase quantities $(f_{j,k})$ will be determined through a game algorithm. However, the ECS can utilize the local user requests $f_{i,j}$ to formulate the contracts.

Specifically, in order to assess the risk brought by the stochastic nature of local requests in future transactions, it is necessary to predict the future contribution of revenue from each local user. Let us denote the number of local users who apply for computing resources from ECS_j as $N_j = \sum_{i=1}^{I} local_{i,j}$. Then, the revenue contribution $Uu_{i,j}^T$ of $User_i$ to ECS_j at time T is represented as:

$$Uu_{i,j}^T = min\left(\frac{L}{N_j}, f_{i,j}^T\right) pl_j - max\left(f_{i,j}^T - \frac{L}{N_j}, 0\right) cp_j \tag{19}$$

Here, ECS_j allocates a computing resource upper limit, L, to $User_i$, where $\frac{L}{N_j}$ represents the average resource upper limit that each participating user can be allocated. $f_{i,j}^T$ denotes the computing resources requested by $User_i$ at time T. The term $min(\frac{L}{N_j}, f_{i,j}^T)pl_j$ represents the revenue that $User_i$ brings to ECS_j at time T, while $max(f_{i,j}^T - \frac{L}{N_j}, 0)cp_j$ represents the compensation that ECS_j pays to $User_i$ at time T. When $f_{i,j}^T < \frac{L}{N_j}$, meaning that the requested computing resources $f_{i,j}^T$ of $User_i$ do not exceed the average resource upper limit, there is no need for compensation, i.e., $max(f_{i,j}^T - \frac{L}{N_j}, 0)cp_j = 0$.

In the contract formulation process, ECS_j assumes that each user can be allocated the requested computing resources, i.e., $L = fl_j^0$. Therefore, the upper limit of computing resources that each user can be allocated, denoted as $\overline{f_j^0}$, is calculated as $\overline{f_j^0} = \frac{fl_j^0}{N_j}$. Based on the value of $f_{i,j}^T$, the requested computing resources of $User_i$ at time T, we can rewrite Equation (19) as:

$$Uu_{i,j}^T = \begin{cases} f_{i,j}^T pl_j, & 0 \le f_{i,j}^T \le \overline{f_j^0} \\ \overline{f_j^0} pl_j - \left(f_{i,j}^T - \overline{f_j^0}\right) cp_j, & \overline{f_j^0} < f_{i,j}^T \end{cases} \tag{20}$$

Furthermore, for ECS_j in the initial state, it cannot predict the specific request quantity $f_{i,j}^T$ in the future. Thus, we represent the computing resource quantity requested by $User_i$ at time T, $E(f_{i,j}^T) = E(Dl_{i,j}^T)/\lambda l_i$, where $E(f_{i,j}^T)$ is the expected value. And $E(Dl_{i,j}^T)$ represents the expected value of the local task request. According to Equations (4) and (6), $\overline{Uu_{i,j}^T}$ can be written as:

$$\overline{Uu_{i,j}^T} = \begin{cases} \frac{E\left(Dl_{i,j}^T\right)}{\lambda l_i} pl_j, & 0 \le \frac{E\left(Dl_{i,j}^T\right)}{\lambda l_i} \le \overline{f_j^0} \\ \overline{f_j^0} pl_j - \left(\frac{E\left(Dl_{i,j}^T\right)}{\lambda l_i} - \overline{f_j^0}\right) cp_j, & \overline{f_j^0} < \frac{E\left(Dl_{i,j}^T\right)}{\lambda l_i} \end{cases} \tag{21}$$

$$\overline{Uu_{i,j}^T} = \begin{cases} \alpha_i \frac{\ln S_{i,j}^0 + \left(u_i - \frac{1}{2}\sigma_i^2\right)T}{\lambda l_i} pl_j, \\ 0 \le \alpha_i \frac{\ln S_{i,j}^0 + \left(u_i - \frac{1}{2}\sigma_i^2\right)T}{\lambda l_i} \le \alpha_i \frac{\overline{\ln S_j^0}}{\lambda l_i} \\ \alpha_i \left(\frac{\overline{\ln S_j^0}}{\lambda l_i} \left(pl_j + cp_j \right) - \frac{\ln S_{i,j}^0 + \left(u_i - \frac{1}{2}\sigma_i^2\right)T}{\lambda l_i} cp_j \right), \\ \alpha_i \frac{\overline{\ln S_j^0}}{\lambda l_i} < \alpha_i \frac{\ln S_{i,j}^0 + \left(u_i - \frac{1}{2}\sigma_i^2\right)T}{\lambda l_i} \end{cases} \tag{22}$$

where the task request quantity α follows a uniform distribution $\alpha_i \sim U(A, B)$.

To mitigate the potential loss risk in each transaction, we incorporate risk assessment into the contract formulation process. We define the risk probability $R_{i,j}$ as the probability that the expected revenue $\overline{Uu_{i,j}^T}$ of ECS_j is less than an acceptable threshold value λu_j. Therefore, the expression for risk probability R_j is given by:

$$R_{i,j} = \Pr\left(\overline{Uu_{i,j}^T} < \lambda u_j \right) \tag{23}$$

A higher revenue risk probability $R_{i,j}$ indicates a greater level of risk for ECS_j from $User_i$. For ease of representation, let us define R as:

$$R = \begin{cases} \frac{\ln S_{i,j}^0 + \left(u_i - \frac{1}{2}\sigma_i^2\right)T}{\lambda l_i} pl_j, 0 \le \ln S_{i,j}^0 + \left(u_i - \frac{1}{2}\sigma_i^2\right)T \le \overline{\ln S_j^0} \\ \frac{\overline{\ln S_j^0}}{\lambda l_i} pl_j - \frac{\overline{\ln S_j^0} - \ln S_{i,j}^0 - \left(u_i - \frac{1}{2}\sigma_i^2\right)T}{\lambda l_i} cp_j, \\ \overline{\ln S_j^0} < \ln S_{i,j}^0 + \left(u_i - \frac{1}{2}\sigma_i^2\right)T \end{cases} \tag{24}$$

Then, the expression for $R_{i,j}$ can be written as:

$$R_{i,j} = \Pr\left(\overline{Uu_{i,j}^T} < \lambda u_j \right) = \begin{cases} 0, & \lambda u_j \le AR \\ \frac{\lambda u_j - AR}{(B-A)R}, & AR < \lambda u_j < BR \\ 1, & BR \le \lambda u_j \end{cases} \tag{25}$$

Finally, the contract L is formulated as:

$$L = \sum_{i=1}^{I} f_{i,j} local_{i,j} Lr_{i,j} \tag{26}$$

Here, $Lr_{i,j}$ is a binary variable indicating whether the risk probability $R_{i,j}$ for ECS_j from $User_i$ satisfies the acceptable risk threshold λr_j. Specifically, when $R_{i,j} > \lambda r_j$, then $Lr_{i,j}^T = 0$; when $R_{i,j} \le \lambda r_j$, then $Lr_{i,j} = 1$.

4. Problem Description

In this section, we describe the transaction details between the ECS and the ENs along with some constraints and challenges they face. We prove that the Stackelberg game subproblem is a strict convex optimization problem and use the Lagrange multiplier method to find the solution to the subproblem for the game. By using backward induction, we obtain a new Stackelberg game upper-level subproblem and prove it to be a strict convex optimization problem as well. By maximizing their respective expected utilities, we determine the resource quantities and prices for the ECSs and the ENs, ultimately obtaining the price update function for the ECS and the request update function for the ENs. Finally, we demonstrate the existence of a Nash equilibrium point in the game.

4.1. Stackelberg Game Model

The Stackelberg game model is used to describe the interactions between a leader and a follower. By the model described in the previous section, the ECS with more edge

computing resources acts as the leader to provide computing resource support to the users and the ENs subscribing to it. The ECS influences the behavior of other participants by adjusting the resource prices. The ENs, on the other hand, are depicted as followers. They have local users subscribing to their computing resources but often need to request some computing resources from the ECS to better complete their services. As followers, the ENs observe the ECS's resource pricing status to decide on the amount of computing resources they would like to purchase.

According to Equation (13), we can describe the profit strategy of the leader ECS as follows:

$$[P1]: \underset{p^T}{argmax} U_E(f,p) = \underset{p^T}{argmax} \sum_{j=1}^{J} Ul_j + Us_j - Cl_j \tag{27}$$

$$s.t. \sum_{j=1}^{J} local_{i,j} \leq 1, \forall i \in I \tag{28}$$

$$0 \leq \|p\|_\infty \leq pmax \tag{29}$$

$$R_{i,j} \leq \lambda r_j, \forall i \in I, j \in J \tag{30}$$

where $f = \{f_{j,k} | j \in J, k \in K\}$ represents the amount of resources allocated to the ENs and $p = \{p_j | j \in J\}$ represents the vector of resource prices. The resource price is limited by an upper bound $pmax$, and $R_{i,j}$ must be less than or equal to an acceptable risk level λr_j for each user and the ECS.

According to Equation (18), we can describe the profit maximization problem for the leader ECS as follows:

$$[P2]: \underset{f}{argmax} U_N(f,p) = \underset{f}{argmax} \sum_{k=1}^{K} Ub_k - Cs_k - Ac_k \tag{31}$$

$$s.t. \sum_{i=1}^{I} local_{i,k} \leq 1, \forall k \in K \tag{32}$$

$$\sum_{j=1}^{J} sub_{j,k} \leq 1, \forall k \in K \tag{33}$$

$$0 \leq \|f\|_\infty \leq fmax \tag{34}$$

$$\sum_{k=1}^{K} f_{j,k} sub_{j,k} \leq max\Big(Q_j - fl_j, Q_j - L_j\Big), \forall j \in J \tag{35}$$

where $f = \{f_{j,k} | j \in J, k \in K\}$ represents the amount of resources allocated to the ENs and $p = \{p_j | j \in J\}$ represents the vector of resource prices. The constraints (34) and (35) impose an upper limit on the amount of computing resources that ENs can request, ensuring that each EN's application does not exceed the allowed limit $fmax$ and the total requested resources by all ENs do not exceed the remaining computing resources of the ECS after delivering to local users.

4.2. Subgame Model Solution

Theorem 1. *$[P2]$ is a strictly convex optimization problem with at most one optimal solution.*

Proof. If the feasible set R is convex and (31) is a strictly concave function with respect to f, then $[P2]$ is a strictly convex optimization problem with at most one optimal solution. The feasible set R is a closed and bounded subset of R, making it convex. (16) and (17) are continuous functions, and (15) is continuous over the feasible set. Hence, (31) is a continuous function with respect to f. To determine the convexity of (31), we examine its

Hessian matrix. Taking the partial derivative of (31) with respect to f, where $f = \{f_{j,k} | j \in J, k \in K\}$:

$$\frac{\partial U_N(f,p)}{\partial f_{j,k}} = \frac{Dtotal_k pl_k sub_{j,k}}{\left(ql_k + f_{j,k} sub_{j,k}\right)^2} - p_j sub_{j,k} - \frac{sub_{j,k}\omega_{j,k}}{W_{j,k}\log\left(1 + P_{j,k}H_{j,k}\right)} \tag{36}$$

$$\frac{\partial^2 U_N(f,p)}{\left(\partial f_{j,k}\right)^2} = -\frac{2Dtotal_k pl_k \left(sub_{j,k}\right)^2}{\left(ql_k + f_{j,k} sub_{j,k}\right)^3} < 0 \tag{37}$$

$$\frac{\partial^2 U_N(f,p)}{\partial f_{j,k}\partial f_{j',k}} = \frac{\partial^2 U_N(f,p)}{\partial f_{j,k}\partial f_{j,k'}} = 0 \tag{38}$$

We observe that the main diagonal elements of the Hessian matrix are negative, which means that the Hessian matrix is negative definite to make the original function be a strictly concave function with respect to f. Thus, Theorem 1 is proven. □

In order to obtain the optimal requested quantity for ENs, they need to obtain the latest resource price p from their leader, ECS. Since ENs cannot directly determine the resource price, they can only follow the edge computing resource prices set by the leader to formulate their purchasing strategy. Therefore, we use the method of Lagrange multipliers to construct the Lagrangian function $L(f,\lambda)$ for $[P2]$. Then, the maximization problem for ENs' maximum revenue becomes:

$$L(f,\lambda) = \max\Big(U_N - \lambda \sum_{j=1}^{J}\sum_{k=1}^{K} f_{j,k}\mathrm{sub}_{j,k} - \max\left(Q_j - fl_j, Q_j - L_j\right)\Big) \tag{39}$$

From this, we construct the dual problem:

$$ming(\lambda) = infL(f,\lambda) \tag{40}$$

$$s.t. \lambda > 0 \tag{41}$$

According to the Karush–Kuhn–Tucker (KKT) conditions, we obtain the optimal solution satisfying the original convex optimization problem $[P2]$, where $f^* = \{f^*_{j,k} | j \in J, k \in K\}$:

$$f^*_{j,k} = \min\left(\max\left(0, \sqrt{\frac{Dtotal_k \cdot pl_k}{p_j + \lambda + \frac{\omega_{j,k}}{W_{j,k}\log\left(1+P_{j,k}H_{j,k}\right)}}} - ql_k\right), fmax\right) \tag{42}$$

Then, we determine the update strategy for λ as:

$$\lambda(t+1) = \lambda(t) + \sigma\frac{\partial g(\lambda)}{\partial \lambda} \tag{43}$$

where σ is the step size determined by using the Armijo method [25].

By backward induction, we substitute the optimal solution f^* determined by ENs at the current value of λ into $[P1]$. It leads to a new problem:

$$\begin{aligned}[P3]: & \underset{p}{argmax} U_E(f^*, p) \\ & = \underset{p}{argmax} \sum_{j=1}^{J} sub_{j,k} \left(\left(Q_j - \sum_{k=1}^{K} f_{j,k}^* \right) pl_j + \right. \\ & \left. \sum_{k=1}^{K} f_{j,k}^* p_j - max \left(fl_j + \sum_{k=1}^{K} f_{j,k}^* - Q_j, 0 \right) cp_j \right)\end{aligned} \tag{44}$$

$$s.t. \sum_{j=1}^{J} local_{i,j} \leq 1, \forall i \in I \tag{45}$$

$$0 \leq \|p\|_\infty \leq Pmax \tag{46}$$

$$R_{i,j} \leq \lambda r_j, \forall i \in I, j \in J \tag{47}$$

where $p = \{p_j | j \in J\}$. By backward induction, after the ECS obtains the resource requests from the follower ENs, it has the knowledge of the total amount of resources to be used. Then, it adjusts the price of computing resources for a new round, allowing the ECS group's revenue to achieve higher expectations.

Theorem 2. *Within a feasible region,* $[P3]$ *is a strictly convex optimization problem.*

Proof. Similar to the proof of Theorem 1, the feasible region is a closed and bounded subset of R, making it a convex set. If $[P3]$ is a strictly concave function with respect to p, then Theorem 2 holds.

In the feasible region, (44) is a continuous function with respect to p. Let us calculate the partial derivative of (44) with respect to p. For convenience, we define $C = \frac{\omega_{j,k}}{W_{j,k} \log(1+P_{j,k}H_{j,k})}$, and we have:

$$\begin{aligned}\frac{\partial U_E}{\partial p_j} = & \sum_{k=1}^{K} sub_{j,k} \left(\sqrt{Dtotal_k pl_k} (p_j + \lambda + C)^{-\frac{1}{2}} \right. \\ & \left. \times \left(1 - \frac{p_j - pl_j - cp_j}{2(p_j + \lambda + C)} \right) - ql_k \right)\end{aligned} \tag{48}$$

$$\begin{aligned}\frac{\partial^2 U_E}{(\partial p_j)^2} = & \sum_{k=1}^{K} sub_{j,k} \sqrt{Dtotal_k pl_k} (p_j + \lambda + C)^{-\frac{3}{2}} \\ & \times \left(\frac{3\left(p_j - pl_j - cp_j\right)}{4(p_j + \lambda + C)} - 1 \right) < 0\end{aligned} \tag{49}$$

$$\frac{\partial^2 U_E}{\partial p_j \partial p_{j'}} = 0 \tag{50}$$

From the Hessian matrix of $[P3]$, it is negative definite, indicating that (44) is a strictly concave function with respect to p. Thus, the proof of Theorem 2 is complete. □

Given the updated function for the quantity of the resources requested by ENs (42), we set $\frac{\partial U_E}{\partial p_j} = 0$ for $[P3]$ within the feasible region and rearrange to obtain the new price updating function:

$$\begin{aligned} p_j^* &= \min\left(\max\left(pl_j + cp_j - \frac{\sum_{k=1}^{K} f_{j,k} sub_{j,k}}{\sum_{k=1}^{K} \frac{\partial f_{j,k}}{\partial p_j} sub_{j,k}}, 0\right), pmax\right) \\ &= \min\Bigg(\max\Bigg(\frac{2\sum_{k=1}^{K} f_{j,k} sub_{j,k}}{\sum_{k=1}^{K} \sqrt{Dtotal_k \cdot pl_k}\,(p_j + \lambda + C)^{-\frac{3}{2}} sub_{j,k}} \\ &\quad + pl_j + cp_j, 0\Bigg), pmax\Bigg) \end{aligned} \tag{51}$$

where $C = \frac{\omega_{j,k}}{W_{j,k}\log(1+P_{j,k}H_{j,k})}$.

4.3. Nash Equilibrium Proof

Based on the problem functions (31) and (44), we construct the standard form of the game:

$$G = N, P, F, U_E, U_N \tag{52}$$

$$N = J \cup K \; and \; J \cap K = \emptyset \tag{53}$$

The strategy spaces P and F for both players are defined as follows:

$$P = \prod_{j \in J} P_j \tag{54}$$

$$F = \prod_{j \in J} \prod_{k \in K} F_{j,k} \tag{55}$$

where P_j and $F_{j,k}$ represent individual strategy sets. P_j represents the strategy set for $\forall j \in J$, with $0 \le p_j \le pmax$. $F_{j,k}$ represents the strategy set for $\forall j \in J, k \in K$, with $0 \le f_{j,k} \le fmax$. Here, p_j and $f_{j,k}$ represent individual action strategies.

The payoff functions U_E and U_N for both players are defined as follows:

$$U_E(p, f) : P \times F \to \mathbb{R}^2, max \sum_{j \in J} Ul_j + Us_j - Cl_j \tag{56}$$

$$U_N(p, f) : P \times F \to \mathbb{R}^2, max \sum_{k \in K} Ub_k - Cs_k - Ac_k \tag{57}$$

where p_j represents the action strategy for leader j, and the strategy vector for all leaders is represented as $p = (p_1, p_2, ..., p_J)$. For $\forall k \in K$, the action strategy chosen by follower k is represented as $f_k = (f_{1,k}, f_{2,k}, ..., f_{J,k})$, and the strategy vector for all followers is represented as $f = (f_1, f_2, ..., f_K)$.

When follower k is given with the leader's strategy vector $p*$, its optimal strategy f_k* is determined as follows:

$$U_N(p^*, f_k^*) = \max_{f_k \in F_k} U_N(p^*, f_k) \tag{58}$$

When leader j is given with the follower's strategy vector $f*$, its optimal strategy p_j* is determined as follows:

$$U_E\left(p_j^*, f^*\right) = \max_{p_j \in P_j} U_E(p_j, f^*) \tag{59}$$

Therefore, if $(p*, f_k*)$ is a Nash equilibrium strategy, then $\exists (p*, f_k*) \in P \times F$ such that (58) and (59) hold. According to the Kakutani fixed-point theorem, we have Theorem 3 as below.

Theorem 3. *If the strategy spaces P and F are non-empty, compact, and convex, and for a given $f*$, for $\forall j \in J$, the function $U_E(p_j, f*)$ is non-empty, compact, and convex on P_j. Similarly, for a*

given $p*$, for $\forall k \in K$, the function $U_N(p, f_k)$ is non-empty, compact, and convex on F_k. Then, this Stackelberg game model G has an equilibrium solution $(p*, f_k*) \in P \times F$.*

Proof. (1) Properties of P and F: It is evident that the strategy spaces P and F are non-empty since both leaders and followers will make at least one action during the game. Since P_j is defined as a closed bounded interval on $\mathbb{R}$, i.e., $P_j = p_j : 0 \leq p_j \leq pmax, j \in J$, it follows that P_j is compact and convex. Similarly, F_k is compact and convex. Since P and F are the Cartesian products of P_j and F_k, respectively, they are compact and convex.

(2) Given a follower's strategy f^*, the optimal strategy set for the leader is non-empty, compact, and convex: Since P is non-empty, for any given follower's strategy f^*, the leader has at least one strategy to choose from, so it is non-empty. For $\forall j \in J$, the function $U_E(p_j, f^*) : P \times F \to \mathbb{R}^2$ is continuous. Therefore, on the compact set P, there must exist a maximum and minimum value, which implies that the optimal strategy set $U_E(p_j, f^*)$ is compact. As U_E is a continuous function, for any two strategies $p_{j_1} \in P_j$ and $p_{j_2} \in P_j$, any linear combination $p_{j_3} = \mu p_{j_1} + (1 - \mu) p_{j_2}$, $\mu \in [0, 1]$, will also belong to this optimal strategy set. That means $U_E(p_{j_1}, f^*) \leq U_E(p_j^*, f^*)$, $U_E(p_{j_2}, f^*) \leq U_E(p_j^*, f^*)$. Then, $U_E(p_{j_3}, f^*) \leq U_E(p_j^*, f^*)$. Hence, the optimal strategy set is convex.

(3) Given a leader's strategy p*, the optimal strategy set for followers is non-empty, compact, and convex. The proof is consistent with (2).

Therefore, this Stackelberg game model G has a Nash equilibrium strategy $(p*, f_k*) \in P \times F$. □

5. Algorithm Analysis

In high-workload environments, the ECS needs to find an optimal strategy for allocating computing resources among its subscribed local users and ENs to maximize resource utilization. At the same time, the ECS-EN game model requires determining suitable resource request quantities and prices. Traditional multi-objective optimization algorithms such as weighted methods and multi-objective genetic algorithms are difficult to apply in this model. Reinforcement learning-based algorithms may result in higher energy consumption and computational delays, which are not suitable for high-workload environments. We need the ECS to assess the risk of transactions and use it as a benchmark for contract signing. The goal is to ensure that the local subscribed users and ENs can avoid issues such as connection failures, transaction failures, inadequate information timeliness, and excessive communication overhead during the contract period. Algorithm 1 is the implementation of the RACA.

Specifically, Algorithm 1 starts by obtaining the subscription relationships $local_{i,j}$, $local_{i,k}$, $sub_{j,k}$ based on the positional information of all ECSs, ENs, and users in the region. By this algorithm, each user can only purchase resources from the ECS or EN they have subscribed to. After obtaining the user's task amount Dl_i under the generalized Wiener process (Step 2), the computing resource requests received by the ECS from its subscribed local users and ENs will be calculated (Steps 3, 4). Each leader ECS then processes the requests it holds based on risk contracts (Steps 6, 9, 13, 16). In particular, when $fen_j < Q_j - L_j$, the contract is considered to be unreasonable, because the follower ENs will not buy more computing resources than they intend. In this case, re-measurement on the risk for the users will be conducted to obtain a new contract L_j* (Steps 17, 18). Using the Stackelberg game, the new computing resource request $\{fen_j*, p_j*\}$ for the ENs at the new contract price p_j* set by the ECS can be obtained.

In the context of contract changes, after obtaining new contracts, both the leader ECS and the follower ENs need to reach a consensus through one round of communication to reach the Nash equilibrium. As we have proven the existence of the Nash equilibrium in this model, it implies that leaders and followers require one round of communication to achieve the Nash equilibrium point. Algorithm 2, presented below, demonstrates the process of finding the Nash equilibrium point through iterative updates of $f_{j,k}$, λ_j, p_j. During the game process, if the current prices remain consistent with the prices from 10 and 20 rounds ago (Step 6), it is assumed that the algorithm has converged to the equilibrium

solution. Simulation experiments have demonstrated the superior performance of the game, with convergence usually achieved within 20 rounds.

Algorithm 1 Risk Assessment Contract Algorithm (RACA).

Input: $I, J, K, S_i, \lambda l_i, Q_j, \lambda u_j, cp_j, pl_j, \lambda r_j, ql_k, pl_k,$
$\omega_{j,k}, W_{j,k}, P_{j,k}, H_{j,k}, E_{j,k}, fmax, pmax, A, B$
Output: Fl_j, Fen_j, Cl_j, L_j
Initialization:
Assign subscription information $local_{i,j}, local_{ik}, sub_{j,k}$ for i, j, k.
User data sets S_i and randomly generates user task request rates α_i from the uniform distribution $[A, B]$.
Initialize prices $p_j \leftarrow cp_j$ and contract parameters $Fl_j, Fen_j, Cl_j, L_j \leftarrow \emptyset$.
$Dl_i \leftarrow \alpha_i, S_i$ // User's task amount.
$fl_j \leftarrow Dl_i, local_{i,j}, \lambda l_i$ // Calculate ECS's local computing resource requests.
$fen_j \leftarrow Dl_i, local_{ik}, pl_k, p_j, ql_k, \omega_{j,k},$
$W_{j,k}, P_{j,k}, H_{j,k}, E_{j,k}, sub_{j,k}$ // Calculate the computing resource requests received by the ECS from ENs.
for $j = 1$ to J **do**
  if $local_{i,j} \cup sub_{j,k} = \emptyset$ **then**
    Go to Step 5 // No subscribed users or ENs.
  end if
  if $local_{i,j} = \emptyset$ **then**
    $Fen_j \leftarrow \min(Fen_j, Q_j)$// No subscribed users but has ENs.
    Go to Step 5
  end if
  if $sub_{j,k} = \emptyset$ **then**
    Go to Step 21 // No subscribed ENs but has users; directly process according to contract.
  end if
  if $fen_j < Q_j - L_j$ **then**
    $R_{i,j} \leftarrow S_i, \lambda l_i, A, B, pl_j, \omega_{j,k}, W_{j,k}, P_{j,k}, H_{j,k},$
    $E_{j,k}, \lambda u_j, local_{i,j}$ // Risk measurement for user.
    $L_j \leftarrow R_{i,j}, \lambda r_j, Dl_i, Q_j, \lambda l_i, local_{i,j}$ // Calculate the new contract.
    $fen_j, p_j \leftarrow$ Calculate the new computing resource request fen_j*, p_j* using Stackelberg game.
  end if
  $Fl_j, Fen_j, Cl_j \leftarrow L_j, fl_j, Q_j, fen_j$ // Actual allocation based on the contract.
  if $Fl_j = Q_j$ **then**
    $Cl_j \leftarrow \emptyset$
  end if
end for

Algorithm 2 Finding Nash equilibrium between ECS and ENs based on convex optimization.

Input: $J, K, Dtotal_k, ql_k, pl_k, fen_j, p_j, cp_j, pl_j, \omega_{j,k}, W_{j,k},$
$P_{j,k}, H_{j,k}, E_{j,k}, sub_{j,k}, fmax, pmax$
Output: $f^*_{j,k}, p^*_j$
Initialization: $\lambda \leftarrow \emptyset$
while Nash equilibrium point not found **do**
  Update resource price $p^*_j(t)$ based on formula (51).
  Update Lagrange multiplier $\lambda^*_j(t)$ based on formula (43).
  if $p(t) = p(t-10)$ and $p(t) = p(t-20)$ **then**
    Exit while loop
  end if
  $t \leftarrow t + 1$
end while

6. Simulation Results

In this section, we evaluate the performance of the Risk Assessment Contract Algorithm (RACA) under the risk contract mechanism for ECS resource allocation and the game algorithm for ECS pricing and EN resource request volume. We provide some explanations. We utilized the publicly available MAWI dataset for Internet traffic tracking trajectories (https://mawi.wide.ad.jp/mawi/ditl/ditl2018-G/, accessed on 20 November 2023). These traces have been demonstrated to exhibit a log-normal distribution [26–28] and comprise IP-level traffic observed in favorable positions within WIDE from 14:00 to 14:15 each day. The traces include anonymized IP and MAC headers. We randomly selected 10 traces as users and divided each trace into 100 time slots, as depicted in Figure 3. Each trace consists of 70 million packets, with an average capture data rate of 422 Mbps. The monitored link capacity is set to 1 Gbps. The number of data generated by users, denoted as S, follows a log-normal distribution, and the variations at the user end nodes are assumed to be Markovian. In a relatively stable system, changes in data volume follow a generalized Wiener process. Most of the traffic remains stable over time. To demonstrate the special case of sudden changes in data traffic, as illustrated by User4 in Figure 3, we ensured the presence of one data stream with high variance among the 10 randomly selected traffic datasets to reflect a more realistic traffic environment.

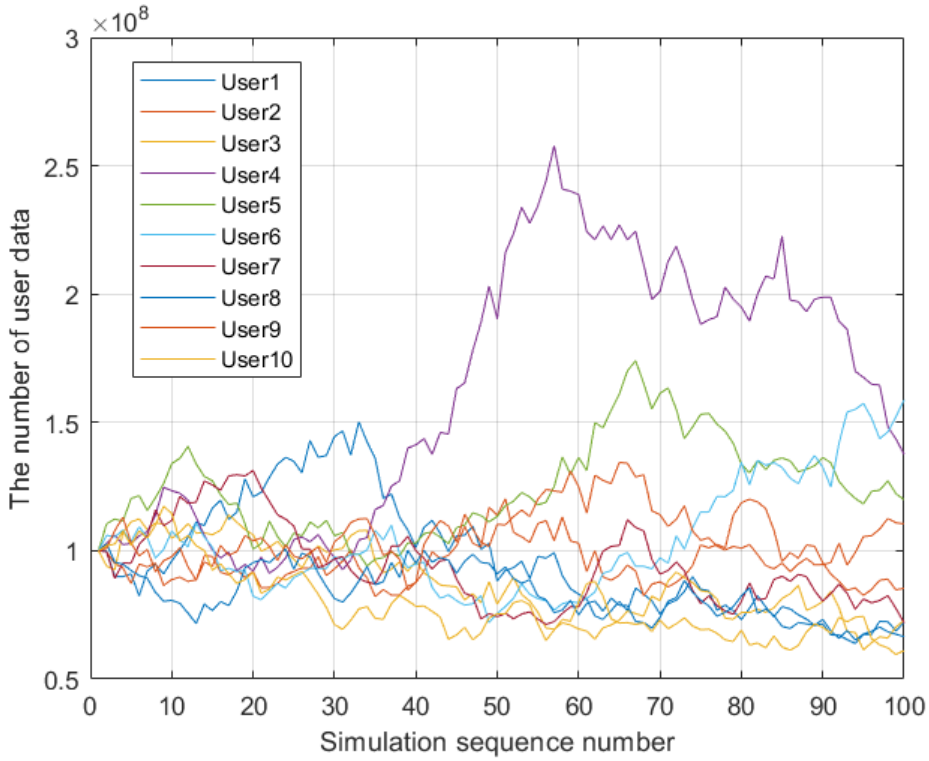


Figure 3. Wiener process data plot of 10 randomly, log-normally distributed datasets divided into 100 time slots.

The location distribution is based on the network topology optimization dataset provided by China Mobile (https://jiutian.10086.cn/open, accessed on 20 November 2023), which includes network topology connectivity data for three cities and element attribute data for a period of time. This dataset contains 20 sets of element data and 1 set of topology connection data. For the topology connection data we used, each piece of location information includes longitude, latitude, and different element capacity values, with an element topology capacity unit of 1 Gbps. The selected distance between each element is less than or equal to 500 m. If the distance were too large, it would not be consistent with the real-world edge environment studied in this paper. By combining the traffic dataset with the element topology relationship data, we reconstructed the real network

environment and topology relationship. We randomly selected 2 ECSs, 3 ENs, and 10 users from these nodes, with all edge devices and user locations randomly distributed within the region. Their binding relationships were determined by the initialization algorithm based on positional relationships. All users were preferentially bound to the nearest EN or ECS and existed as subscribers to them. The computing resources of edge devices were consistent with the capacity of the network elements. Communication was conducted using WiFi technology, with data referenced from the documents [23,29–35]. Refer to Table 3 for specific parameter details.

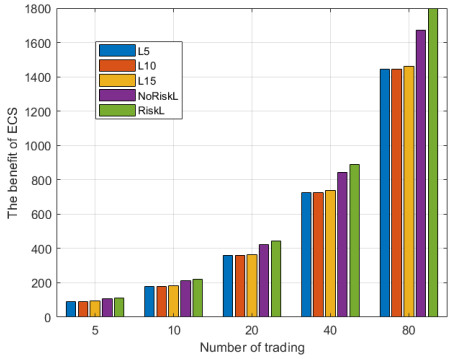
Table 3. Main simulation parameters.

Parameter	Value
Expected task completion time λl_j	1 s
Transmission bandwidth	30 Mhz
Transmit power	150 mW
Channel gain	5 DB
ECS computing resource Q_j	20G CPU cycles/s
ECS compensation price cp_j	1
ECS local task unit value pl_j	1
Acceptable risk threshold λr_j	0.5
EN computing resource ql_k	4G CPU cycles/s
EN task value per second pl_k	5
Delay loss value $C_{j,k}$	1
Resource purchase and pricing upper limits f_{max}, p_{max}	[99 G CPU cycles/s, 50]
Task request rate limits A, B	[0.2; 0.6]

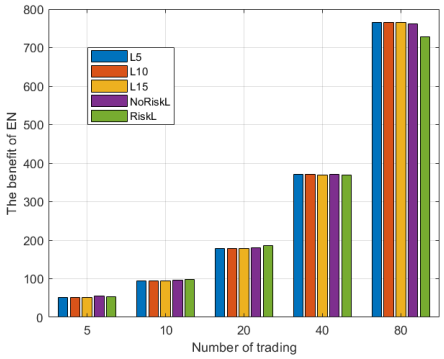
6.1. Performance Evaluation of Risk Assessment

In the edge contracts proposed by us, risk assessment plays a crucial role. The risk assessment for each edge node directly affects the updating of the contract value L. Different contract values L represent different allocation methods faced by ECSs when handling resource allocation. Therefore, we describe the impact of risk assessment on edge contracts by examining the system's final revenue and energy consumption. To investigate the impact of risk assessment on the system, we designed several methods to obtain the contract value L: three fixed contract values (L5, L10, L15), obtaining the contract value L without risk assessment (NoRiskL), and obtaining the contract value L through risk assessment (RiskL) as proposed by us. The fixed contract values (L5, L10, L15) were determined according to the proportion of ECS computing resources. Obtaining the contract value L without risk assessment (NoRiskL) refers to dividing the ECS computing resource Q based on the ratio of the computing resources requested by each user and EN. All methods are based on game algorithms, and the subsequent allocation of EN resources adopts a proportional allocation method. The results are shown in Figures 4–6.

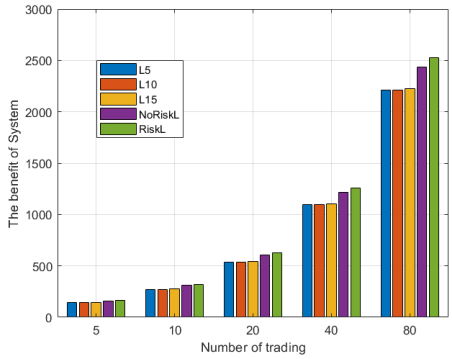
We compared the total revenue of the ECS (Figure 4a), total revenue of the ENs (Figure 4b), and total revenue of the system (Figure 4c) with the number of transactions. Here, the total revenue of the system represents the sum of the total revenue of the ECS and EN. In Figure 4a, we can observe that with the increase in the number of transactions, the edge contracts with risk assessment consistently outperform others in ECS revenue. Figure 4b demonstrates that with the increase in the number of transactions, due to the limitation of local users' access to computing resources, the methods with fixed contract values perform better in the revenue part of EN. This indicates that edge contracts with risk assessment are more inclined towards the revenue performance of EN. On the other hand, as shown in Figure 4c, in terms of the total revenue of the system, dynamic contract values L perform better than fixed contract values L. Moreover, the contract values obtained through risk assessment outperform the contract values allocated proportionally in terms of the total revenue of the system.



(**a**) The benefit of the ECS



(**b**) The benefit of the EN



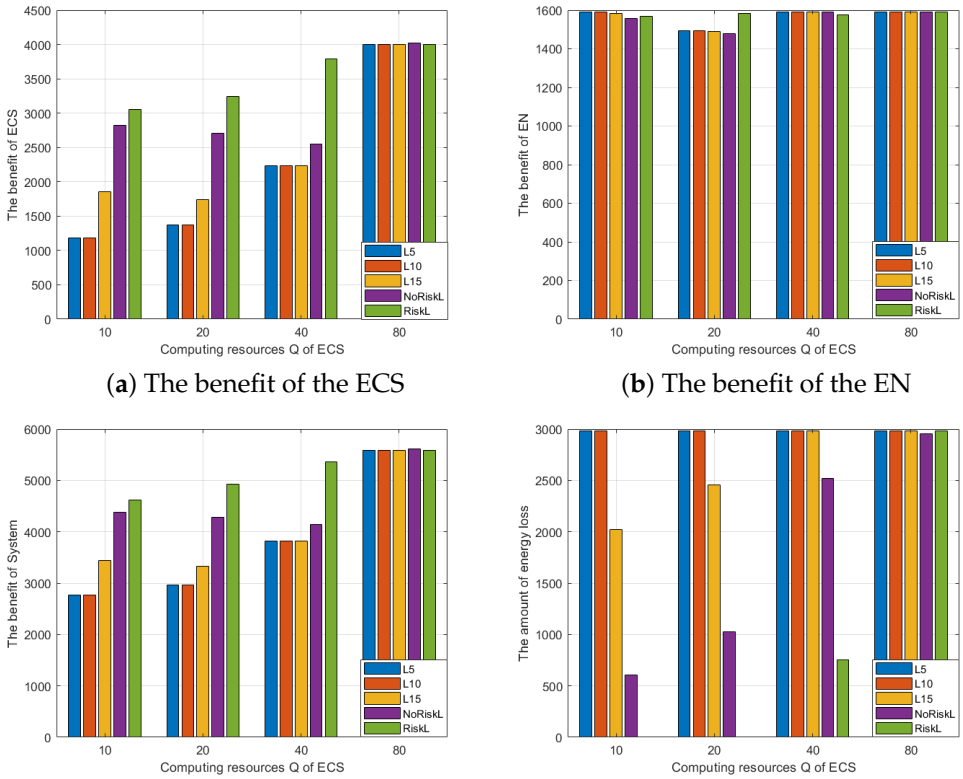
(**c**) The benefit of the system

Figure 4. Comparison and evaluation of risk assessment under different transaction times.

Figure 5 depicts the performance comparison evaluation of risk assessment and other methods under different ECS computing resources Q. From Figure 5a–c, we can observe that under conditions of insufficient ECS computing resources or high load in the edge resource pool, risk assessment can effectively improve the revenue of the ECS and the total revenue of the system. In Figure 5a, as ECS computing resources gradually become sufficient, we observe that the performance of risk assessment and other methods in ECS revenue begin to converge, indicating that the risk assessment method performs better under high load conditions. Figure 5b shows that as the computing resources Q increase, the impact on the total revenue of the EN is not significant, as the contract mechanism itself ensures the EN's resource acquisition capability. In terms of the total revenue of the system (Figure 5c), risk assessment demonstrates better performance in assisting the system to achieve revenue effects under various computing resource scenarios.

For edge devices, limited energy is a significant characteristic distinguishing them from traditional cloud devices. Using algorithms with lower computational consumption implies that edge devices can have longer battery life in adaptive environments. In this section, we mainly investigate the energy consumption of the three methods discussed in the previous section during computational allocation. Because it is difficult to quantify the additional energy expenditure and battery consumption during the transaction process (for example, estimating the battery consumption for each quotation), we use the number of game rounds to describe it [20]. As shown in Figure 5d, under the condition of other parameters being unchanged, increasing the computing resource capacity Q of the ECS will also increase the energy consumption of the risk assessment method until, eventually, the energy consumption of all methods becomes consistent. When ECS computing resources are low or the load is particularly high, dynamic contract values are more energy-saving than static contract values. Among fixed contract values, L15 is more energy-efficient than other fixed contract values, indicating that an appropriate contract value L can effectively reduce resource waste. The risk assessment method has lower energy consumption compared to

other methods, indicating that contract values obtained through risk assessment can help ECSs reduce energy consumption during allocation, eliminating the process of ECS and EN bargaining to find the best allocation method. When the ECS's computing resources increase or it is in a low-load situation and the ECS's computing resources are sufficient to meet all nodes, the ECS increases the number of game rounds to pursue better performance, and energy consumption tends to be consistent.

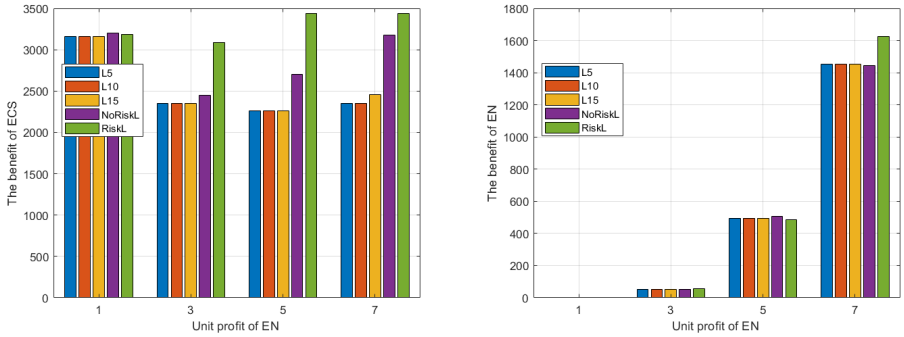


(**a**) The benefit of the ECS (**b**) The benefit of the EN

(**c**) The benefit of the system (**d**) The amount of energy loss

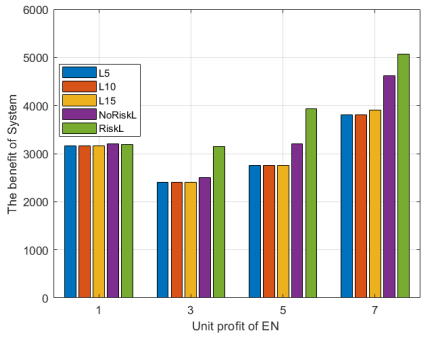
Figure 5. Comparison and evaluation of risk assessment under different ECS computing resources.

Figure 6 illustrates the performance of the ECS, the EN, the system, and the losses under different EN profit efficiencies in the risk assessment method. As shown in Figure 6b, with the increase in the EN's unit profit, the system tends to allocate more computing resources to the EN. There is no significant difference between the risk assessment method and other methods in terms of the EN's profit, but as shown in Figure 6a,c, the risk assessment method outperforms other methods in terms of the ECS's profit and overall system profit. In Figure 6d, as the EN's profit increases, the energy loss under fixed contract values also increases, but in some cases, the energy loss may decrease, such as the energy loss of L15, which first increases and then decreases. This is because fixed contract values cannot adapt to the changing EN conditions, and different EN unit profits should have different contract values to adapt. This leads to the possibility of both increasing and decreasing energy losses under fixed contract values. However, the RiskL and NoRiskL methods formulate contract values more suitable for different EN unit profits. Finally, we can also see that the risk assessment method significantly reduces energy consumption compared to all other methods.

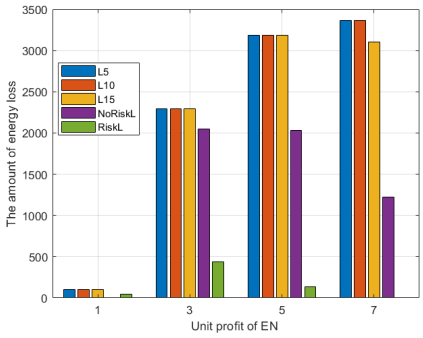


(**a**) The benefit of the ECS

(**b**) The benefit of the EN



(**c**) The benefit of the system



(**d**) The amount of energy loss

Figure 6. Comparison and evaluation of risk assessment under different EN unit profits.

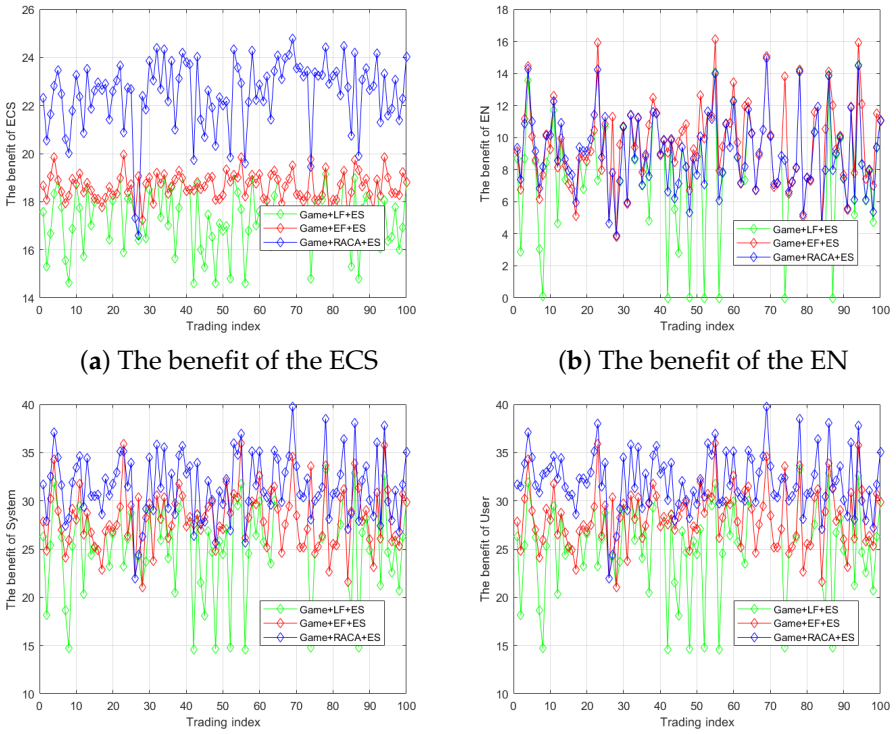
6.2. Performance Evaluation of RACA

In these experiments, the amount of computing resources purchased by ENs ($f_{j,k}$) from the ECS and the price of computing resources (p_j) are determined by a game-theoretic approach, which is employed by the ENs to allocate computing resources, which are proportionally distributed, to their subscribed local users. Each game incurs its associated delays and energy consumption. After determining $f_{j,k}$ and p_j for the ECS and ENs, to address the resource allocation problem between the ECS's local users and the ENs, we compare RACA with some of the latest studies incorporating incentive mechanism designs. However, the models and problems in these works differ from ours, making direct comparisons challenging. Therefore, we tailored the basic ideas of these algorithms and carefully designed three comparative mechanisms: GAME + Risk Assessment Contract Algorithm (RACA) + Equal Share (ES), GAME + Local First (LF) + ES, and GAME + EN First (EN) + ES. The results are shown in Figures 7–9.

Figure 7 illustrates the distribution of earnings for the three algorithms over 100 transactions. We compare the ECS's total revenue (Figure 7a), the EN's total revenue (Figure 7b), the system's total revenue (Figure 7c), and the total revenue of users (Figure 7d). The total revenue of users consists of three components: revenue from the computational resources provided by the ECS, compensation revenue from the ECS, and revenue from the computational resources provided by the EN. In Figure 7a,b, we observe that RACA outperforms other methods in ECS revenue, while its performance in EN revenue is mostly consistent with the other two methods, with RACA occasionally falling short compared to EF. It can be seen that RACA is a distribution method advantageous to the ECS, reducing the energy and latency costs incurred during the game, enabling the ECS to make faster decisions on how to allocate computational resources to local users and ENs to maximize ECS's revenue. In Figure 7c, the system's total revenue demonstrates a significant advantage for RACA. In Figure 7d, although users incur losses exceeding the contract value in RACA, the ECS provides some compensation to users, and users at ENs receive more revenue. Therefore, RACA still performs well in terms of user revenue.

As shown in Figure 8a–c, under unchanged conditions, increasing the computational resource capacity Q of ECSs will enhance the revenue of both ECSs and ENs. With the continuous increase in Q, the revenue capability of the RACA algorithm will gradually become comparable to that of LF and EF. This is because when an ECS has sufficient computational resources to meet the demands of all ENs and user nodes, the revenue capabilities of RACA become equivalent. However, in low-computational-resource or high-load scenarios, RACA demonstrates superior performance in terms of ECS and system revenue capabilities. Figure 8d illustrates the energy consumption of the three algorithms under the influence of the ECS's computational resources. In scenarios of computational resource scarcity or high loads, the energy consumption of RACA may decrease to zero. This indicates that in high-load scenarios, RACA tends to directly utilize contracts for transactions to avoid further resource waste. However, when an ECS's computational resources increase or the reduced demands from ENs and users allow the ECS to respond adequately, RACA may increase the number of games played to pursue superior outcomes.

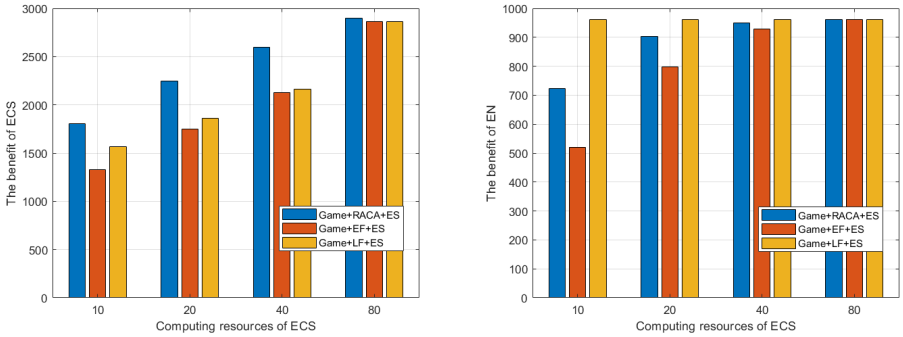
Figure 9 illustrates the performance of the ECS, the EN, the system, and losses under RACA for different EN unit profits. As shown in Figure 9a,c, RACA demonstrates superior performance in terms of ECS revenue and system revenue. Figure 9b shows that as the unit profit of ENs increases, the total revenue of ENs also increases. However, when the unit profit of ENs is too small, ENs incur no revenue under all three methods. This is because the unit profit of ENs is too small, resulting in ENs being unwilling to request computational resources from ECSs, as the revenue obtained from requesting resources from ECSs is lower than the communication cost. This also explains the phenomenon observed in Figure 9a, where the total revenue of ECSs increases as the unit profit of ENs decreases. Figure 9d reflects the significant advantage of RACA in reducing energy consumption. Overall, RACA demonstrates superior performance across different unit profits of ENs.



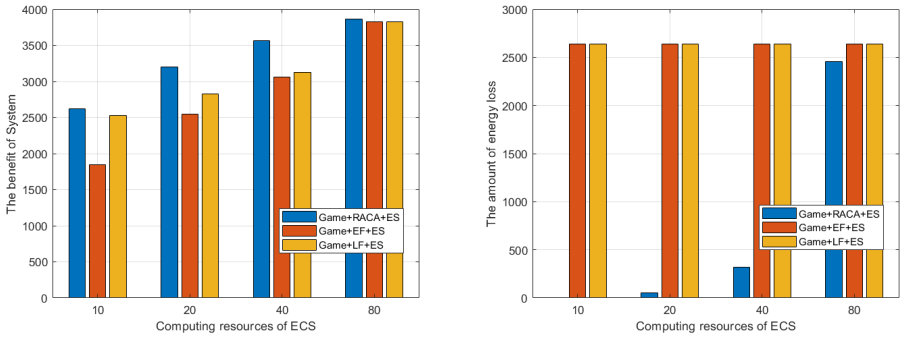
(**a**) The benefit of the ECS (**b**) The benefit of the EN

(**c**) The benefit of the system (**d**) The benefit of the User

Figure 7. Comparison and evaluation of RACA under 100 transactions.

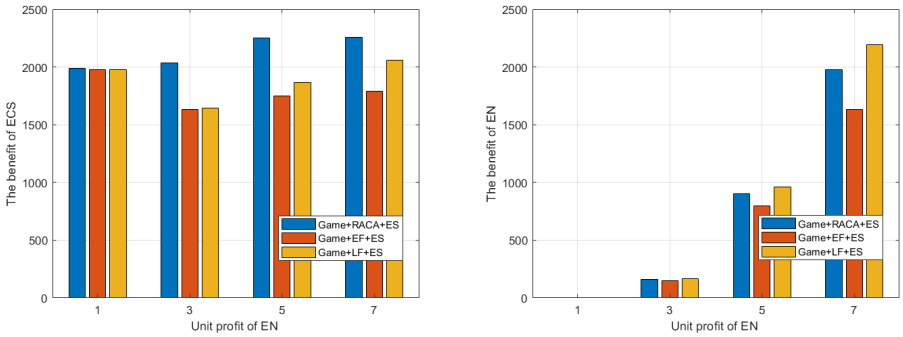


(**a**) The benefit of the ECS (**b**) The benefit of the EN

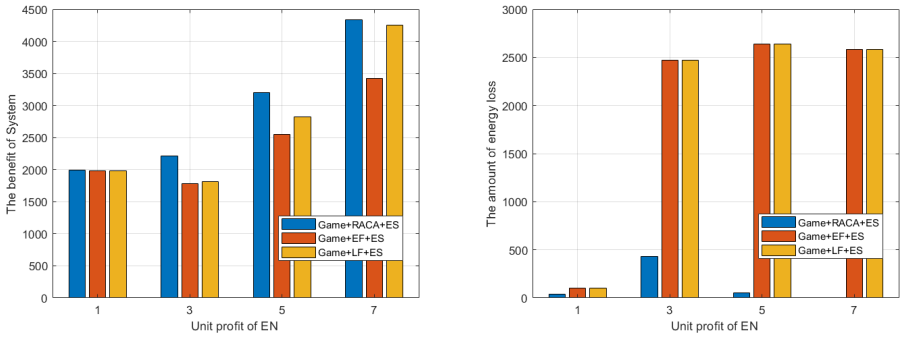


(**c**) The benefit of the system (**d**) The amount of energy loss

Figure 8. Comparison and evaluation of RACA under different ECS computing resources.



(**a**) The benefit of the ECS (**b**) The benefit of the EN



(**c**) The benefit of the system (**d**) The amount of energy loss

Figure 9. Comparison and evaluation of RACA under different different EN unit profits.

6.3. Comparison of EN Resource Allocation

The computing resources allocated to the ENs need to be further distributed to their users. In this experiment, each EN has a Nash equilibrium solution obtained by the game

convergence for resource allocation among its local users. We compare this equilibrium-based proportional allocation (ES) with other allocation methods. Specifically, we compare the ES scheme with the following schemes, including the Equal Share (ES), the User Equal Distribution (ED), the Buyer Quality First (BQF), the Buyer Number Most (BNM), the Sequential Allocation (SA), and the Random Allocation (RA) schemes. In the BQF scheme, priority is given to allocating resources to high-quality users with more requests, while the BNM prioritizes users with fewer requests to benefit more users overall.

The results in Figure 10 show that when the computing resources purchased by the ENs are equal to the requested resources, the ES, the BQF, the BNM, and the SA schemes can achieve consistent profits, outperforming the ED and the RA schemes in terms of revenue. When the purchased computing resources are less than the requested resources, the ES outperforms all other allocation methods due to its reliance on the equilibrium solution obtained from the game. It demonstrates the superiority of using the equilibrium solution obtained by the game algorithm and distributing resources according to the game ratio.

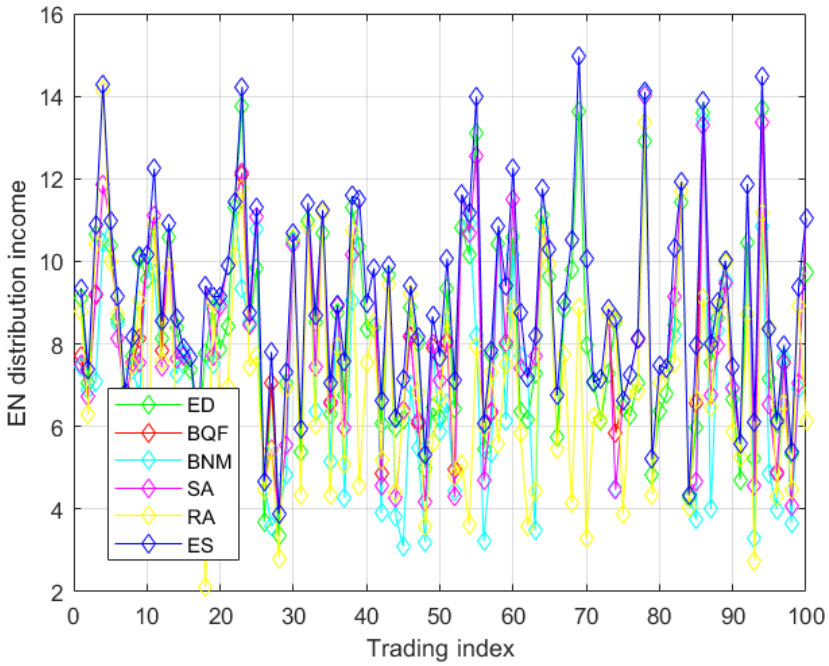


Figure 10. Comparison and evaluation of EN resource allocation under 100 transactions.

7. Conclusions and Future Work

In this paper, we have addressed the resource allocation problem in edge computing under high-workload conditions by using risk-based edge contracts and game algorithms based on the Lagrange multiplier method. Specifically, we have formulated a resource pool model for multiple ECSs and ENs using the Stackelberg game model to maximize the overall revenue. This problem has been decomposed into two subproblems: ECS computing resource allocation and ECS pricing along with ENs' resource requests. We have estimated the risk for all local users subscribed to the ECS and have used edge contracts to solve the first subproblem. We have employed the game algorithm based on the Lagrange multiplier method to find the solution to the second subproblem.

We have also conducted theoretical analysis to prove the convexity of both the upper level and lower level of the master–slave game and established the Nash equilibrium solutions. We have developed the RACA algorithm and game algorithm to solve the problems involved. Simulation results demonstrated the superiority of the proposed scheme under high-workload conditions.

The contribution of this work lies in the proposal of edge contracts based on risk assessment metrics. By incorporating the concept of edge risk into edge traffic based

on risk assessment theory from investment markets, it introduces a new perspective for addressing profit relationships between edge devices using a market-based model. While the edge devices autonomously update contract values, the Stackelberg game is used to determine edge system resource pricing and task offloading, providing a more dynamic and effective solution for resource allocation in edge environments. The autonomous update of contracts based on risk assessment not only increases the overall revenue of the edge system but also reduces energy consumption, with more significant effects in high-load or low-computing-resource environments, addressing the challenge of resource scarcity in high-load edge environments.

To enable the application of this work on a broader platform, further research will be conducted on the model. The edge resource pricing in this paper adopts a uniform pricing method, where the pricing of edge resources owned by the same ECS is the same. However, in real life, resource providers often have differentiated pricing for resources (such as network operators pricing traffic differently for different users). Therefore, future analysis will examine the impact of ECSs' differential pricing for different ENs on the edge system. Additionally, although the data prediction in this paper is based on existing log-normal distribution models, real-world networks are often more complex and variable. Thus, there is a need to further improve the accuracy of network traffic prediction to enhance its generalization and adaptability. For example, combining neural networks with other methods to predict traffic changes can make contract values more accurate. Lastly, we hope to expand edge user task scheduling beyond the EN or ECS they subscribe to. Through graph theory models, we aim to establish a more decentralized many-to-many matching scheduling object model, enabling the model to be applied in decentralized complex environments, such as the autonomous control of drone clusters.

Author Contributions: Conceptualization, M.S.; Methodology, M.S. and M.M.; Software, M.S. and Y.S.; Validation, M.S.; Formal analysis, M.S.; Investigation, Y.S.; Resources, H.W.; Data curation, M.S.; Writing—original draft, M.S.; Writing—review and editing, H.W., M.M., and R.Z.; Visualization, M.S.; Supervision, H.W., M.M., and R.Z.; Project administration, H.W.; Funding acquisition, H.W. All authors have read and agreed to the published version of the manuscript.

Funding: This This research was funded by Zhejiang Normal University, grant number 62171413, 2022-4-063.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. The Derivation of *dDl* Using the Ito Process

As Dl is a function of S and t, based on the Taylor expansion, we have:

$$dDl = \frac{\partial Dl}{\partial S}dS + \frac{\partial Dl}{\partial t}dt + \frac{1}{2}\frac{\partial^2 G}{\partial S^2}(dS)^2 + \frac{1}{2}\frac{\partial^2 G}{\partial t^2}(dt)^2 + \frac{\partial^2 G}{\partial S \partial t}dSdt + \ldots \tag{A1}$$

Using Equation (1), we obtain:

$$(dS)^2 = u^2S^2(dt)^2 + 2u\sigma S^2(dt)^{\frac{3}{2}} + \sigma^2 S^2 \epsilon^2 dt \tag{A2}$$

Substituting Equations (1) and (A2), and considering that $(dt)^2$, $(dt)^{\frac{3}{2}}$, and $dSdt$ are higher-order infinitesimals of dt, we can rewrite Equation (A1) as:

$$\begin{aligned} dDl &= \frac{\partial Dl}{\partial S}(uSdt + \sigma Sdz) + \frac{\partial Dl}{\partial t}dt + \frac{1}{2}\frac{\partial^2 G}{\partial S^2}\sigma^2 S^2 \epsilon^2 dt \\ &= \frac{\alpha}{S}(uSdt + \sigma Sdz) + 0 \cdot dt + \frac{1}{2}\left(-\frac{\alpha}{S^2}\right)\sigma^2 S^2 \epsilon^2 dt \\ &= \alpha\left(u - \frac{1}{2}\sigma^2\epsilon^2\right)dt + \alpha\sigma dz \end{aligned} \tag{A3}$$

As $\epsilon \sim N(0,1)$, the mean of $E(\epsilon^2) = 1$. Thus, Equation (A3) can be simplified to:

$$dDl = \alpha\left(u - \frac{1}{2}\sigma^2\right)dt + \alpha\sigma dz \tag{A4}$$

From Equation (A4), it is evident that Dl also follows a Wiener process.

References

1. Chen, X.; Jiao, L.; Li, W.; Fu, X. Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing. *IEEE/ACM Trans. Netw.* **2016**, *24*, 2795–2808. [CrossRef]
2. Yi, C.; Cai, J.; Su, Z. A multi-user mobile computation offloading and transmission scheduling mechanism for delay-sensitive applications. *IEEE Trans. Mob. Comput.* **2019**, *19*, 29–43. [CrossRef]
3. Chiang, M.; Zhang, T. Fog and IoT: An overview of research opportunities. *IEEE Internet Things J.* **2016**, *3*, 854–864. [CrossRef]
4. Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K.B. A survey on mobile edge computing: The communication perspective. *IEEE Commun. Surv. Tutorials* **2017**, *19*, 2322–2358. [CrossRef]
5. Bozkaya, E. A Digital Twin Framework for Edge Server Placement in Mobile Edge Computing. In Proceedings of the International Informatics and Software Engineering Conference (IISEC), Ankara, Turkey, 21–22 December 2023; Volume 4, pp. 1–6.
6. Luo, R.; Jin, H.; He, Q.; Wu, S.; Xia, X. Enabling Balanced Data Deduplication in Mobile Edge Computing. *IEEE Trans. Parallel Distrib. Syst.* **2023**, *34*, 1420–1431. [CrossRef]
7. Yuan, L.; He, Q.; Tan, S.; Li, B.; Yu, J.; Chen, F.; Yang, Y. CoopEdge+: Enabling Decentralized, Secure and Cooperative Multi-Access Edge Computing Based on Blockchain. *IEEE Trans. Parallel Distrib. Syst.* **2023**, *34*, 894–908. [CrossRef]
8. Dai, P.; Hu, K.; Wu, X.; Xing, H.; Teng, F.; Yu, Z. A probabilistic approach for cooperative computation offloading in MEC-assisted vehicular networks. *IEEE Trans. Intell. Transp. Syst.* **2020**, *23*, 899–911. [CrossRef]
9. Jamil, M.N.; Hossain, M.S.; Islam, R.U.; Andersson, K. Workload Orchestration in Multi-Access Edge Computing Using Belief Rule-Based Approach. *IEEE Access* **2023**, *11*, 118002–118023. [CrossRef]
10. Khazali, A.; Bozorgchenani, A.; Tarchi, D.; Shayesteh, M.G.; Kalbkhani, H. Joint Task Assignment, Power Allocation and Node Grouping for Cooperative Computing in NOMA-mmWave Mobile Edge Computing. *IEEE Access* **2023**, *11*, 93664–93678. [CrossRef]
11. Hu, B.; Gao, Y.; Zhang, W.; Jia, D.; Liu, H. Computation Offloading and Resource Allocation in IoT-Based Mobile Edge Computing Systems. In Proceedings of the 2023 IEEE International Conference on Smart Internet of Things (SmartIoT), Xining, China, 25–27 August 2023; pp. 119–123.
12. Song, C.; Xu, W.; Wu, T.; Yu, S.; Zeng, P.; Zhang, N. QoE-driven edge caching in vehicle networks based on deep reinforcement learning. *IEEE Trans. Veh. Technol.* **2021**, *70*, 5286–5295. [CrossRef]
13. Cheng, Y.; Liang, C.; Chen, Q.; Yu, F.R. Energy-efficient D2D-assisted computation offloading in NOMA-enabled cognitive networks. *IEEE Trans. Veh. Technol.* **2021**, *70*, 13441–13446. [CrossRef]
14. Hu, X.; Tang, X.; Yu, Y.; Qiu, S.; Chen, S. Joint load balancing and offloading optimization in multiple parked vehicle-assisted edge computing. *Wirel. Commun. Mob. Comput.* **2021**, *2021*, 1–13. [CrossRef]
15. Wang, J.; Liu, K.; Li, B.; Liu, T.; Li, R.; Han, Z. Delay-sensitive multi-period computation offloading with reliability guarantees in fog networks. *IEEE Trans. Mob. Comput.* **2020**, *19*, 2062–2075. [CrossRef]
16. Chen, R.; Li, L.; Hou, R.; Yang, T.; Wang, L.; Pan, M. Data-driven optimization for resource provision in non-cooperative edge computing market. In Proceedings of the ICC 2020—2020 IEEE International Conference on Communications (ICC), Virtually, 7–11 June 2020; pp. 1–6.
17. Iyer, G.N.; Raman, V.; Aswin, K.; Veeravalli, B. On the strategies for risk aware cloud and fog broker arbitrage mechanisms. In Proceedings of the 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC),Erode, India, 11–13 March 2020; Volume 4, pp. 794–799.
18. Peng, K.; Huang, H.; Liu, P.; Xu, X.; Leung, V.C. Joint optimization of energy conservation and privacy preservation for intelligent task offloading in mec-enabled smart cities. *IEEE Trans. Green Commun. Netw.* **2022**, *6*, 1671–1682. [CrossRef]
19. Du, J.; Cheng, W.; Lu, G.; Cao, H.; Chu, X.; Zhang, Z.; Wang, J. Resource pricing and allocation in MEC enabled blockchain systems: An A3C deep reinforcement learning approach. *IEEE Trans. Netw. Sci. Eng.* **2021**, *9*, 33–44. [CrossRef]

20. Liwang, M.; Gao, Z.; Wang, X. Let's trade in the future! A futures-enabled fast resource trading mechanism in edge computing-assisted UAV networks. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 3252–3270. [CrossRef]
21. Marbukh, V. Towards Robust Fog/Edge Computing Infrastructure with Risk Adjusted Multi-Connectivity. In Proceedings of the 2022 9th International Conference on Future Internet of Things and Cloud (FiCloud), Rome, Italy, 22–24 August 2022; Volume 9, pp. 161–166.
22. Yi, C.; Huang, S.; Cai, J. Joint resource allocation for device-to-device communication assisted fog computing. *IEEE Trans. Mob. Comput.* **2019**, *20*, 1076–1091. [CrossRef]
23. Dai, P.; Hu, K.; Wu, X.; Xing, H.; Yu, Z. Asynchronous deep reinforcement learning for data-driven task offloading in MEC-empowered vehicular networks. In Proceedings of the IEEE INFOCOM 2021-IEEE Conference on Computer Communications, Vancouver, BC, Canada, 10–13 May 2021; pp. 1–10.
24. Wu, S.; Shi, Z. Itôwave: Itô stochastic differential equation is all you need for wave generation. In Proceedings of the ICASSP 2022—2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, 22–27 May 2022; pp. 8422–8426.
25. Fujiwara, K.; Okamoto, Y.; Kameari, A.; Ahagon, A. The Newton-Raphson method accelerated by using a line search-comparison between energy functional and residual minimization. *IEEE Trans. Magn.* **2005**, *41*, 1724–1727. [CrossRef]
26. Mawi Archive. 2004. Available online: http://mawi.wide.ad.jp/ (accessed on 20 November 2023).
27. Alasmar, M.; Parisis, G.; Clegg, R.G.; Zakhleniuk, N. On the Distribution of Traffic Volumes in the Internet and its Implications. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 955–963.
28. Alasmar, M.; Clegg, R.; Zakhleniuk, N.; Parisis, G. Internet Traffic Volumes Are Not Gaussian—They Are Log-Normal: An 18-Year Longitudinal Study With Implications for Modelling and Prediction. *IEEE/ACM Trans. Netw.* **2021**, *29*, 1266–1279. [CrossRef]
29. Dai, Y.; Xu, D.; Maharjan, S.; Zhang, Y. Joint load balancing and offloading in vehicular edge computing and networks. *IEEE Internet Things J.* **2018**, *6*, 4377–4387. [CrossRef]
30. You, C.; Huang, K.; Chae, H.; Kim, B.H. Energy-efficient resource allocation for mobile-edge computation offloading. *IEEE Trans. Wirel. Commun.* **2016**, *16*, 1397–1411. [CrossRef]
31. Du, J.; Yu, F.R.; Chu, X.; Feng, J.; Lu, G. Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization. *IEEE Trans. Veh. Technol.* **2018**, *68*, 1079–1092. [CrossRef]
32. Chen, M.; Hao, Y. Task offloading for mobile edge computing in software defined ultra-dense network. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 587–597. [CrossRef]
33. Liu, Y.; Yu, H.; Xie, S.; Zhang, Y. Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks. *IEEE Trans. Veh. Technol.* **2019**, *68*, 11158–11168. [CrossRef]
34. Qi, Q.; Wang, J.; Ma, Z.; Sun, H.; Cao, Y.; Zhang, L.; Liao, J. Knowledge-driven service offloading decision for vehicular edge computing: A deep reinforcement learning approach. *IEEE Trans. Veh. Technol.* **2019**, *68*, 4192–4203. [CrossRef]
35. Ning, Z.; Dong, P.; Wang, X.; Obaidat, M.S.; Hu, X.; Guo, L.; Guo, Y.; Huang, J.; Hu, B.; Li, Y. When deep reinforcement learning meets 5G-enabled vehicular networks: A distributed offloading framework for traffic big data. *IEEE Trans. Ind. Informatics* **2019**, *16*, 1352–1361. [CrossRef]