mathematics

MDPI

Article

Incentive Mechanism for Cloud Service Offloading in Edge–Cloud Computing Environment

Chendie Yao *, Junjie Xie and Zhong Liu

Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha 410073, China; xiejunjie@nudt.edu.cn (J.X.); liuzhong@nudt.edu.cn (Z.L.)
* Correspondence: yaochendie@nudt.edu.cn

Abstract: Edge computing refers to provision storage and computation resources at the network edge, closer to end users than the remote cloud. In such edge–cloud computing environments, many cloud providers intend to offload cloud services to the edge nodes to offer high-quality services for data-intensive and latency-sensitive applications. The major obstacle is that edge nodes are rarely willing to offer resources voluntarily without any rewards. To this end, this paper proposes an efficient incentive mechanism for edge–cloud computing environments using Stackelberg game theory to motivate more edge nodes to host offloaded cloud services. We analyze the properties of the game model and present a solution to compute the unique Stackelberg Equilibrium (SE) of the nonlinear model. On this basis, we propose an efficient polynomial-time algorithm to find the SE. Moreover, we discuss the adaptation of our incentive mechanism to dynamic node joining or departing. Performance evaluations compare our incentive mechanism with three benchmarks and a state-of-the-art mechanism. The results indicate that our incentive mechanism can effectively motivate both the edge nodes and the remote cloud to participate in the edge–cloud environment, achieving maximum resource utilization with minimal rewards while remaining robust in dynamic situations.

Keywords: incentive mechanism; edge–cloud computing; offloaded cloud service; Stackelberg game

MSC: 91-01

check for updates

Academic Editors: Daniel-Ioan Curiac and David Carfì

Received: 16 March 2025
Revised: 24 April 2025
Accepted: 27 April 2025
Published: 21 May 2025

Citation: Yao, C.; Xie, J.; Liu, Z. Incentive Mechanism for Cloud Service Offloading in Edge–Cloud Computing Environment. *Mathematics* **2025**, *13*, 1685. https://doi.org/10.3390/math13101685

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In the traditional cloud computing paradigm, the cloud environment is responsible for serving the requests of various users. Recently, the emergence of novel data-intensive and time-sensitive applications poses significant challenges to the network capacity and service delay of cloud computing. For example, augmented reality (AR) and video analytics generate tremendous data, which are transported to the remote cloud environment [1]. Consequently, edge computing [2] has been proposed as a promising way to serve those data-intensive applications at the network edge instead of the remote cloud environment in a timely manner.

Edge computing tends to shift the cloud service to the edge nodes. An edge node is an alliance of a set of computing servers deployed close to end users, which has less storage and computing resources than a cloud node [3–6]. Inspired by the benefits of edge computing, cloud nodes also intend to offload their services to suitable edge nodes [7], which have spare available resources to host such cloud services. In this way, edge nodes undertake the tasks of handling requests of large-scale end users, which are processed by the cloud

node originally. The end users may obtain required services at the network edge directly, replacing remote cloud storage service. This kind of cloud service offloading system would significantly shorten service latency and ease network traffic pressure. For example, many edge-assisted video streaming frameworks are proposed to offload the video enhancement technology from the cloud node to edge nodes [8–12]. In the conventional video streaming framework, video enhancement technologies, which can improve the video quality, are generally implemented at the cloud node. However, with the persistent pursuit of higher-resolution and -quality videos, the cloud node may not meet the low transmission delay requirement, thus degrading the user quality of experience (QoE). Additionally, it cannot satisfy user needs' uncertainty and variability to preprocess and store massive video files in the cloud for various personalized video services. Aided with emerging edge computing technologies, video enhancement services are expected to migrate from the cloud node to the network edge to address this problem.

Due to its great potential benefits, edge–cloud computing has attracted many research efforts from both academia and industry. Research findings include the mechanisms and algorithms to (1) maximize the number of requests processed by the edge nodes and the utility of resources at the network edge by jointly considering service placement and request scheduling in the edge–cloud computing environment [8–10,13,14]; (2) improve the quality of service (QoS) at the network edge and minimize the service delay [15–19]; and (3) maintain fair caching balance among different edge providers [20,21]. Other researchers propose constructing an efficient data storage system at the network edge [22] or realizing image feature extraction at the network edge while ensuring data privacy [23].

Prior research proposals have two main common assumptions: (1) there exist a given number of edge nodes that are willing to host the offloaded service from various cloud platforms, and (2) such edge nodes can offer sufficient resources to support the placement of those offloaded services and serve users' requests for such services. Thus, for a cloud service released as a new task to the network edge, the cloud node needs to negotiate with all willing edge nodes, and finally selects some edge nodes to accommodate its offloaded service. These two consumptions, however, face serious challenges. Each edge node would consume considerable resources to take over any offloaded service from a cloud node. This would decrease the resource costs used to deploy its local edge services. Hence, edge nodes lack the motivation to help cloud nodes unless they receive a satisfying reward.

It remains open to construct an edge–cloud computing environment where sufficient edge nodes are willing to contribute partial resources to serve more offloaded cloud services. In this paper, we design an effective incentive mechanism to realize such a desired edge–cloud computing environment. That is, edge nodes prefer to determine the amount of resources allocated to accommodate an offloaded cloud service, according to the reward the corresponding cloud node provides. Due to the heterogeneity of edge nodes, there are significant differences in the resource cost budget of edge nodes, making designing such incentive mechanisms more challenging.

In this paper, we model the assignment of offloaded cloud tasks as a Stackelberg game [24], based on which we introduce an incentive mechanism to motivate the cloud nodes and more edge nodes to join the edge–cloud computing environment. In this mechanism, cloud nodes provide a fixed reward for all involved edge nodes and such edge nodes obtain payment according to their contribution to the offloaded cloud task. Note that we consider the heterogeneity of edge nodes and the limited resource cost budget of each edge node. These two factors complicate our problem and cause great challenges in solving the problem. In particular, node heterogeneity introduces individual differences, nonlinear constraints, and dynamic policy dependencies, which significantly increase the complexity of modeling and solving.

We design an incentive mechanism based on the Stackelberg game theory and present a detailed process to compute the unique Stackelberg Equilibrium (SE). With the SE, the cloud utility is maximized and none of the edge nodes can increase their utility via changing their strategy. Next, we present an efficient algorithm for obtaining SE point(s) within polynomial time. Extensive experiments show that the proposed mechanism achieves excellent results compared to three benchmarks and the state-of-the-art mechanism. The proposed incentive mechanism can motivate edge nodes effectively with a relatively low reward. It is also robust to realistic dynamic situations. To the best of our knowledge, our work is the first to propose an incentive mechanism that focuses on both the competition between the cloud node and edge nodes and the internal competition among edge nodes for the edge–cloud computing environment.

The remainder of this paper is organized as follows. We discuss related work in Section 2 and illustrate the offloaded cloud task problem and our system model in Section 3. We then design the incentive mechanism and report the optimal strategies in Section 4. In Section 5, we show our algorithm according to the optimal strategy of Section 4. Then, we rethink the incentive mechanism for the dynamic edge–cloud computing environment in Section 6. We evaluate the performance of our incentive mechanism and compare it with the benchmarks and the state-of-the-art mechanism in Section 7 and show a discussion in Section 8. Finally, we conclude this paper in Section 9.

2. Related Work

2.1. Service Offloading in Mobile Cloud Computing

In recent years, mobile cloud offloading has received a significant amount of attention as it brings the benefits of cloud computing to mobile devices. Due to limited computation ability and energy consumption, mobile devices (e.g., cellphones and laptops) are unable to handle new mobile application requests that require high quality of service (QoS) and a high level of processing [25]. A suitable solution for processing these requests is mobile cloud offloading [26], which offloads application services from user equipment to the conventional centralized cloud. CloneCloud is a flexible runtime partitioning approach to combine static analysis and dynamic profiling and seamlessly offload part of the application execution from mobile devices to a computational cloud [11]. The work in [12] studied the multi-user computation partitioning problem, which combined the partitioning of multiple users' computations and the scheduling of offloaded computations to the cloud resources to optimize the average completion time for all the users.

The work in [27] proposed a game-theoretic approach for the multi-user computation offloading problem where each user had to decide whether to offload. The work in [28] proposed a multi-resource allocation problem in the cloudlet-based environment considering resource-intensive and latency-sensitive mobile applications. This problem aimed to maximize the long-term reward and was formulated as a semi-Markov decision process. MCloud is a context-aware code offloading framework that considers a multi-tier offloading resource environment, including the mobile ad hoc network, cloudlet, and public clouds [29]. The work in [30] proposed an elastic computation framework that consisted of resource allocation component, task-scheduling algorithm, and task dispatch middleware. This framework also considered a multi-tier computation resource environment. However, mobile cloud offloading introduces significant extra execution delay consisting of the transmission time of the offloaded applications to the cloud and the backhaul delay. Such a delay makes mobile cloud offloading unsuitable for real-time applications.

2.2. Edge Computing

To deal with the extra delay of mobile cloud offloading, a new computing paradigm, edge computing, is introduced. Edge computing brings storage and computation resources from the centralized cloud to distributed edge nodes close to end users, thus speeding up the request processing rate and reducing the load of the cloud node. Researchers have proposed many technologies for edge computing [2].

Some works focused on joint decision-making of service placement to maximize the number of requests processed by the edge nodes or the utility of resources at the network edge [8,10,13,31]. For instance, Ref. [10] proposed the joint optimization of service placement and request routing to store non-trivial amounts of data and meet asymmetric bandwidth requirements in mobile-edge multi-cell networks. The work in [8,13] studied the optimal provisioning of edge services via joint service placement and request scheduling. Considering the critical effects of access network selection on the network congestion, the work in [31] jointly optimized the network selection and service placement to improve the quality of service (QoS) of user applications.

Another set of research about edge computing concerns task scheduling and resource allocation in the edge–cloud environment. Dedas [32] studied online deadline-aware task dispatching and scheduling in the edge–cloud environment, aiming to maximize the number of finished tasks within the deadline. Hetero-Edge [15] is an approach to resource allocation and orchestration for real-time response demanding based on inherently distributed and heterogeneous edge nodes. Pairwise Markov Chain [23] is a task-scheduling strategy for privacy-preserving image feature extraction on edge nodes.

Additionally, research about edge computing includes many other aspects. For instance, the work in [20,21] concerned caching fairness in the edge environment and formulated the caching load balance problem as an integer linear programming problem. Coin [22] proposed constructing an efficient data storage system at the network edge. EdgeEDR [33] is an online market mechanism for energy emergency demand response in the edge–cloud environment via cloudlet control. However, most of these works assume that edge nodes consume their resources and host the service offloaded from the cloud voluntarily with little consideration of the selfishness and the rationality of the edge node.

2.3. Incentive Mechanism in Edge Computing

Several economics theories have been applied to design incentive mechanisms in edge computing, including the Stackelberg game, market model, and contract theory. RMCS [34] focuses on mobile crowdsensing and involves an incentive-compatible active participant recruitment scheme via deep learning. The work in [35] studied mobile crowdsensing as well. It motivated the fog nodes to return their resource sensing data to the fog cloud via the Stackelberg game in fog computing. In [36], the author designed an efficient incentive mechanism for edge caching in large-scale mobile edge networks. Aiming to maximize the utility of the task publisher, a contract-based incentive mechanism was designed for computation offloading in fog computing in [37]. The work in [38] designed an incentive mechanism for computation offloading. It used the Stackelberg game to model the competition between the cloud node and the edge node and presented a multi-round algorithm to search for the optimal solution.

Refs. [37,38] mentioned above designed the incentive mechanism for computation offloading in edge computing with one study using the Stackelberg game. However, they just considered the negotiation between the cloud node and edge nodes but ignored the competition among edge nodes. Meanwhile, none of them provided a closed-form optimal solution for the two-stage Stackelberg game with constraints. In this paper, we focus on how to motivate edge nodes to participate in the offloaded cloud task with consideration

of the two-stage competitions (i.e., the competition between the cloud and edge nodes, and that among edge nodes). We use the Stackelberg game to model the problem and present a fast and efficient algorithm to obtain the closed-form optimal solution.

3. System Model and Problem Formulation

In this section, we first illustrate the definition of the offloaded cloud task and how the offloaded cloud task works. Next, we show our model for the task. Finally, we analyze the challenges and propose the method to find the optimal analytic solution to the task model.

3.1. Problem Description

Edge computing prefers to deploy edge nodes close to the end users with limited storage and computation resources than cloud computing, to serve users' requests in a timely manner. In this paper, we consider an edge–cloud computing system, where the edge nodes may undertake services originally hosted by some remote cloud nodes. We define an offloaded cloud task as an event that a cloud service needs to be placed at the network edge. Remote cloud nodes intend to offload those popular services to some edge nodes. This can significantly improve the QoS by processing requests from nearby users at the cost of consuming resources of the edge nodes.

Figure 1a depicts a general framework of the edge–cloud network, consisting of the cloud nodes, the edge nodes, and the end users. The set of edge nodes, which has an interest in providing resources to accommodate the offloaded cloud tasks, is denoted as $E = \{1, 2, \ldots, k\} (k \geq 2)$. Figure 1b shows the workflow of an edge–cloud computing system. A cloud node first releases one offloaded cloud task. If an edge node is willing to perform this task, it will consume additional resources, which are originally used to deploy its own local services. That is, an edge node participating in hosting offloaded cloud service would incur a cost; hence, it would expect a payment from the corresponding cloud node. Given an offloaded cloud task, each edge node will make a service providing plan according to the cost as well as the return, including the amount of allocated storage and computation resources, and then submit it to the related cloud node. After collecting the service providing plans from sufficient edge nodes, the cloud node will select a part of the participated edge nodes, denoted as $S = \{1, 2, \ldots, n\} (n \geq 2)$. If those selected edge nodes have offered resources to carry out the offloaded cloud tasks, the cloud node computes and sends the payment for each selected edge node.

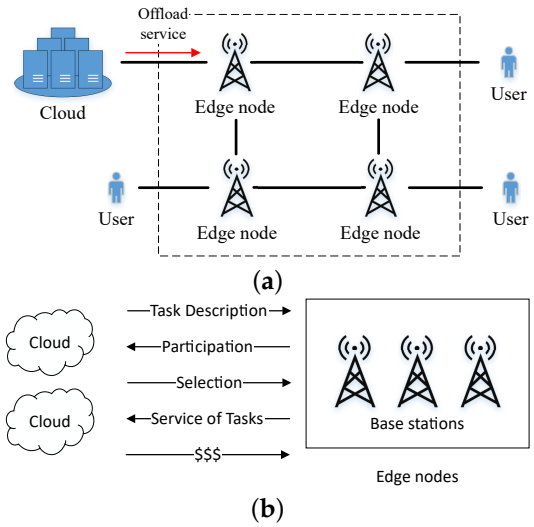


Figure 1. Illustration of an edge–cloud computing system. (**a**) Edge–cloud network structure; (**b**) edge–cloud computing system workflow.

During the whole process of the offloaded cloud task, the cloud node aims to maximize its own utility. Those edge nodes are rational and independent since they are operated by different edge providers. Therefore, each edge node also desires to maximize its own utility and accommodate an offloaded cloud task under sufficient incentives. Table 1 lists some important notations and terms used in the remainder of this paper.

Table 1. Terms and notations.

Notation	Description
N	the set of edge nodes
S	the set of selected edge nodes
n	the number of selected edge nodes
R	the reward offered by a cloud node
u_i	the utility of edge node i
u_0	the utility of a cloud node
γ	the weight of the resource cost in u_0
c_i	the resource cost of edge node i
M_i	the resource cost budget of edge node i
M_{max}	the upper bound of resource cost budget among edge nodes

3.2. Model of Offloaded Cloud Task

Given an offloaded cloud task, the cloud node announces a reward $R > 0$, to motivate potential edge nodes to compete for hosting this task. Then, each edge node decides the level of participation willing after comparing the reward and the resource cost due to accommodating that task.

Let c_i denote the cost of the resources that edge node i consumes to serve an offloaded cloud task, including the costs for storage, computation, and communication resources. R represents the reward that the cloud node offers to all the edge nodes.

Assume that each edge node is rewarded according to the proportion of its contribution to the offloaded cloud task. The contribution of edge node i is measured by the resource amount it allocates to the task. We set each edge node to have the same unit cost as the resource. Thus, the contribution of edge node i can be represented by the ratio of its own resource cost c_i to the total costs of all the edge nodes. The reward edge node i receives $\frac{c_i}{\sum j \in S c_j} R$. The more resource cost an edge node contributes, the higher reward from the cloud node it receives. For example, a node will receive 5% of the announced reward R if it contributes 5% of the total resource cost the cloud node has collected, and 10% of R if providing 10% of the total resource cost. Considering that the edge node consumes resource costs for the task, the utility of edge node i (i.e., the net profit) is defined as the received reward minus the caused resource cost. We have

$$u_i = \frac{c_i}{\sum_{j \in S} c_j} R - c_i. \tag{1}$$

We define the utility of the corresponding cloud node as the value of the offloaded cloud task minus the reward this cloud node pays. The value of the task is measured by the caused resource cost these participated edge nodes paid for this task. Then, we calculate the cloud node utility as follows:

$$u_0 = \gamma \sum_{i \in S} c_i - R, \tag{2}$$

where $\gamma > 1$ is the weight of the resource cost paid by the participating edge nodes.

Note that each edge node has a limited resource cost budget. We impose a constraint on the resource cost edge node i can afford for hosting that offloaded cloud service as

$$0 \leq c_i \leq M_i, \tag{3}$$

where M_i denotes the resource cost budget of edge node i. The utility functions of edge node i as well as the cloud node, u_i and u_0, have to satisfy this constraint.

In this paper, we use the Stackelberg game to model the competition between the cloud node and edge nodes. Meanwhile, we also notice the internal competition among edge nodes. Rational and selfish, each edge node desires to maximize its own utility (the reward it obtains minus the resource cost it pays) in this edge–cloud system. This desire will hurt other edge nodes' interests since the total reward they can receive is finite. Most studies that used the Stackelberg game did not consider internal competition [38,39]. In these studies, the reward of one edge node (i.e., one follower) would not be affected by another one (i.e., another follower's decision) directly. That is, one edge node's decision c_i would affect the value of the reward R offered by the cloud node, and then the value of the reward affects another edge node's decision c_j.

In our incentive mechanism, we propose that each edge node is rewarded according to the proportion of its contribution to the offloaded cloud task as shown in Formula (1). Hence, the edge node would impact other edge nodes' utility directly via its resource cost decision. This idea imposes the internal competition of edge nodes and is more realistic than the traditional Stackelberg game.

Additionally, considering the limited resource cost budget, we impose a constraint on the limited resource cost budget edge node i can afford as shown in Formula (3). This constraint makes our incentive mechanism more realistic since no edge node has infinite resource costs. The incentive mechanism with the constraint of the limited resource cost is different from the original model of the Stackelberg game. It is because of this constraint that we face new challenges when solving the optimal solution as illustrated in Section 3.3.

3.3. Challenges and Overview of the Solving Process

To develop a large-scale edge–cloud computing environment, it is very crucial to motivate edge providers to construct more edge nodes that are willing to cooperate with those remote cloud nodes. The obstacle is that an edge node is usually unwilling to cooperate unless the received reward from the corresponding cloud nodes outperforms the consumed resource cost. Although many proposals have been developed to promote the paradigm of edge computing, few of them study the incentive mechanism for the edge–cloud computing environment.

For an offloaded cloud task, we propose the joint processing model between cloud nodes and edge nodes in Section 3.2. Accordingly, we propose using the Stackelberg game theory to model the incentive mechanism, where a cloud node acts as the leader while edge nodes act as the followers. The leader and the followers only concentrate on maximizing their own utility. The solution for the cloud node is defined as the reward strategy, i.e., the reward it is willing to pay for the offloaded task. For an edge node, the solution is defined as the resource strategy, i.e. the cost of the resource it provides for the offloaded cloud task. Hence, the optimal solution of this game is defined as a set of stable strategies, where no player, i.e., the cloud node or the edge node, can increase its utility by changing its current strategy. It is very complicated to find the optimal game strategies, due to the following two challenges in our scenario.

First, the utility of an edge node is nonlinear and is correlated to the behaviors of other edge nodes. Hence, the optimal game solution directly tends to be correlated with other edge nodes' optimal solutions. We have to analyze the relationship between this edge node itself and other edge nodes and then try to give the final analytic solutions. Second, the constraints, c_i and R, restrict the space of feasible solutions. The utility functions (1) and (2)

are not strictly concave about c_i and R. Thus, we cannot solve the optimal game solution via the derivative of the utility function directly.

The Stackelberg game has been utilized for solving the networking problems, such as crowdsourcing sensing [40] exchanging files in peer-to-peer networks [39], and computation offloading in edge computing [38]. However, existing solutions are not suitable to solve our problem because of the above two challenges. For example, Yang et al. did not consider the serious impact of constraints in [40]. Liu et al. ignored the co-relationship among the followers while just considering the game between the leader and the followers in [38,39].

Before applying the KKT conditions, we verify their applicability to our problem. Although the utility function in Equation (1) is nonlinear and fractional, it is continuously differentiable with respect to c_i over the feasible region. The constraint set defined by Equation (3) is convex, and the Slater condition is satisfied because there exist strictly feasible points $(0 \leq c_i \leq M_i)$ for the problem.

The fractional structure in Equations (1) and (2) introduces mathematical complexity to our optimization approach. This structure creates a coupling effect between decision variables of different edge nodes through the term $\frac{c_i}{\sum_{j \in S} c_j}$, resulting in a non-convex objective function. Consequently, KKT conditions alone cannot guarantee global optimality. To overcome this challenge, we implement a methodical piecewise analysis strategy by partitioning the solution space into three distinct regions based on the values of R and M_i (as later detailed in Cases 1–3). Within each region, we apply KKT conditions to derive necessary conditions for optimality, then systematically analyze these conditions across all regions and their boundaries to construct a complete characterization of the optimal solution. Through this comprehensive analysis, we verify that our derived solution satisfies the first-order optimality conditions throughout the entire feasible region, thereby confirming that we have obtained the global optimum despite the non-convexity challenges. This approach enables us to address the difficulties posed by the fractional and nonlinear utility functions and derive the complete solution as demonstrated in subsequent theorems and propositions.

In the following sections, we will illustrate how to find the analytic equation of the optimal strategies in detail. We first present our incentive mechanism model with the Stackelberg game and introduce the definition of Stackelberg Equilibrium (SE). To obtain the SE, we need to find the optimal strategies of the followers, i.e., the optimal resource strategies for the edge nodes, and accordingly use the optimal resource strategies to solve the optimal strategy of the leader, i.e., the optimal reward strategy, for the cloud node in turn. To find the optimal resource strategies, we use the Karush–Kuhn–Tucker (KKT) conditions to obtain the preliminary solution and then analyze the relationship between edge nodes to find the optimal resource strategies. Accordingly, we discuss the optimal reward strategy for the cloud node in several different cases.

Additionally, we rethink the solution of our problem in the dynamic edge–cloud environment, where new edge nodes join or existing edge nodes leave on demand. We analyze how to find the new optimal strategies for both the cloud node and edge nodes when an edge node joins or leaves the edge–cloud computing environment.

4. Stackelberg Game-Based Incentive Mechanism

In this paper, we model the incentive mechanism for hosting an offloaded cloud task as a Stackelberg game. This is a strategic game, where one player acts as the leader while other players act as followers to compete for resources. Given an offloaded cloud task, there are two stages in the Stackelberg game. First, the related cloud node broadcasts its reward R to all potential edge nodes. Second, each willing edge node designs its resource allocation strategy for accommodating such offloaded cloud tasks while maximizing its own utility. In this paper, we define the cloud node that offloads tasks as the leader and

those edge nodes that host such a task as the followers. The strategy of the cloud node is to determine its reward R for its offloaded task. The strategy of edge node i is to identify its resource cost c_i for serving that task. We denote the strategy vector consisting of all edge nodes as $\mathbf{c} = (c_1, c_2, \ldots, c_n)$. Note that we consider the second stage of the Stackelberg game as a non-cooperative game that we call the Resource Determination game.

Our incentive mechanism based on the Stackelberg game can better motivate the edge nodes to participate in this task compared with other strategies. The objective of this game is to find the Stackelberg Equilibrium (SE) where no player has an incentive to deviate from his chosen strategy after considering an opponent's choice (explained in detail in Section 4.1). In the SE, each player's strategy is optimal considering the decisions of other players. Every player will win since everyone obtains the outcome they desire [24]. Thus, the rational and selfish edge nodes would be willing to participate in the task under the Stackelberg game-based incentive mechanism.

Ignoring the rationality and selfishness of the edge nodes, other strategies, e.g., assigning the offloaded cloud service from a cloud node to specific or random edge nodes with a reward, cannot attract as many edge nodes as our proposal. First, in such strategies, the allocation can hardly satisfy all the edge nodes. For example, once the cloud node assigns the task demand unevenly, those edge nodes who receive low resource demands will obtain less reward than those who receive high resource demands. Then, the lower ones might be unwilling to participate in the task because of the unfairness. Second, the cloud node would have difficulty in setting the reward. The cloud node is baffled if all the edge nodes collude to argue a high reward that exceeds the reward tolerance of the cloud node. The cloud node has to raise the reward, which hurts its own interest; otherwise, its task cannot be accomplished.

4.1. Stackelberg Equilibrium

The Stackelberg game consists of the Pricing Reward game for the cloud node and the Resource Determination game for the edge nodes. The objective of the cloud node is to maximize its utility after releasing an offloaded cloud task, while the objective of each edge node is to maximize its own utility after accommodating that offloaded cloud task. Then, the objective of the whole game is to find the Stackelberg Equilibrium (SE) point(s), where each optimal strategy is the best reply to all other optimal strategies stated in the literature [24]. The SE of our game problem is defined as follows.

Definition 1. *Let R^* denote a strategy for the Pricing Reward game of the cloud node, while c_i^* denotes a strategy for the Resource Determination game of edge node i. Then, we have a strategy vector $\mathbf{c} = (c_1, c_2, \ldots, c_n)$. The point $(R^*, \mathbf{c}^*)$ is an SE if the following conditions are satisfied for any $(R, \mathbf{c})$ with $R \geq 0$ and $\mathbf{c} \succeq 0$:*

$$u_0(R^*, \mathbf{c}^*) \geq u_0(R, \mathbf{c}^*) \tag{4}$$

$$u_i(c_i^*, R^*) \geq u_i(c_i, R^*) \tag{5}$$

where u_0 and u_i are the utility of the cloud node and edge node i, respectively, defined as Formulas (1) and (2).

For our Stackelberg game, the SE can be achieved via two steps. First, given the reward R of a cloud node, we solve the Resource Determination game to obtain the best resource response $\mathbf{c}^*$ from edge nodes. Second, with the $\mathbf{c}^*$ we obtained, we solve the Pricing Reward game to achieve the optimal reward R^*. We will introduce the details of these two steps in Section 4.2 and Section 4.3, respectively.

4.2. Optimal Resource Strategy

When the cloud node reports its reward R, the optimal resource strategy c_i^* for edge node i can be determined.

Theorem 1. *Given the reward R offered by a cloud node, the optimal solution for the Resource Determination game is given by*

$$\forall i \in S,\ k = 1, 2, \cdots, K,$$
$$c_i^* = \begin{cases} 0, & R \le \alpha_i \\ \sqrt{R\alpha_i} - \alpha_i, & \alpha_i < R \le \frac{(M_i+\alpha_i)^2}{\alpha_i} \\ M_i, & R > \frac{(M_i+\alpha_i)^2}{\alpha_i} \end{cases}, \tag{6}$$

where $\alpha_i = \sum_{j \in S \setminus \{i\}} c_j^*$.

Proof. The Resource Determination game of the edge node i can be written as

$$P1 : max\ \frac{c_i}{\sum_{j \in S} c_j} R - c_i, \tag{7}$$
$$s.t. \quad c_i \le M_i, \tag{8}$$
$$c_i \ge 0, \tag{9}$$

where the objective of edge node i is to maximize its own utility. Edge node i can increase its own utility by changing its resource cost from 0 to M_i.

The Lagrangian function of $P1$ can be written as

$$L(c_i, \lambda, \mu) = c_i - \frac{c_i}{\sum_{j \in S} c_j} R + \lambda(c_i - M_i) - \mu c_i \tag{10}$$

where λ and μ are the non-negative dual variables with the constraints mentioned in the literature [41].

The dual function is $g(\lambda, \mu) = sup_{c_i} L(c_i, \lambda, \mu)$. Then, we have the Lagrangian dual problem $min_{\lambda \ge 0, \mu \ge 0}\ g(\lambda, \mu)$.

The duality gap is zero for the convex problem; hence, we can solve the original problem by finding the optimal solution of its dual problem. To realize the zero-gap results between these two problems, the optimal solution needs to satisfy the following KKT conditions [39,42]:

$$\lambda \ge 0, \mu \ge 0, \tag{11}$$
$$\lambda(c_i^* - M_i) = 0 \tag{12}$$
$$\mu c_i^* = 0 \tag{13}$$
$$0 \le c_i^* \le M_i \tag{14}$$
$$\frac{\partial L(c_i^*, \lambda, \mu)}{\partial c_i^*} = 0 \tag{15}$$

Formula (15) follows $\frac{\partial L(c_i^*, \lambda, \mu)}{\partial c_i^*} = 1 + \frac{c_i^* R}{(\sum_{j \in S} c_j)^2} - \frac{R}{\sum_{j \in S} c_j} + \lambda - \mu = 0$. Then, we have

$$c_i^* = \sqrt{\frac{R \sum_{j \in S \setminus \{i\}} c_j}{1 + \lambda - \mu}} - \sum_{j \in S \setminus \{i\}} c_j \tag{16}$$

Suppose $c_i^* < M_i$ when $R > \sqrt{\frac{(M_i+\sum_{j\in S\setminus\{i\}} c_j)^2}{\sum_{j\in S\setminus\{i\}} c_j}}$. From Formula (12), we have $\lambda = 0$. Therefore, Formula (16) reduces to $c_i^* = \sqrt{\frac{R\sum_{j\in S\setminus\{i\}} c_j}{1-\mu}} - \sum_{j\in S\setminus\{i\}} c_j$. Then, $c_i^* < M_i$ results in $\frac{R\sum_{j\in S\setminus\{i\}} c_j}{1-\mu} < (M_i + \sum_{j\in S\setminus\{i\}} c_j)^2$. Since $\mu \geq 0$, we get $R < \frac{(M_i+\sum_{j\in S\setminus\{i\}} c_j)^2}{\sum_{j\in S\setminus\{i\}} c_j}$. This contradicts the presumption $c_i^* < M_i$ when $R > \sqrt{\frac{(M_i+\sum_{j\in S\setminus\{i\}} c_j)^2}{\sum_{j\in S\setminus\{i\}} c_j}}$. Therefore, according to Formula (12), we get

$$c_i^* = M_i, \text{ if } R > \frac{(M_i + \sum_{j\in S\setminus\{i\}} c_j)^2}{\sum_{j\in S\setminus\{i\}} c_j} \tag{17}$$

Similarly, we prove that $c_i^* = 0$, if $R \leq \sum_{j\in S\setminus\{i\}} c_j$, and $c_i^* = \sqrt{R\sum_{j\in S\setminus\{i\}} c_j} - \sum_{j\in S\setminus\{i\}} c_j$, if $\sum_{j\in S\setminus\{i\}} c_j < R \leq \frac{(M_i+\sum_{j\in S\setminus\{i\}} c_j)^2}{\sum_{j\in S\setminus\{i\}} c_j}$. Let α_i denote the sum of the resource cost of all edge nodes except edge node i, i.e., $\alpha_i = \sum_{j\in S\setminus\{i\}} c_j^*$. So far, we have achieved the preliminary optimal resource strategy for each edge node as shown in Formula (6). Thus, Theorem 1 is proved. □

Remark 1. *We can observe from Formula (6) that c_i^* is a piecewise function about the variable R. In general, c_i^* is an increasing function of R. If R is in the low level, the optimal resource strategy c_i^* for edge node i is 0. If R is in the middle level, edge node i will provide its fractional resource for accommodating that offload cloud task. If R is in the high level, edge node i will provide all of its resources M_i to do so. This indicates that the cloud node can motivate the edge nodes to provide more resources by improving its reward R. It is also noticed that c_i^* is influenced by other edge nodes due to the introduction of α_i in Formula (6). Thus, we need to conduct more efforts to obtain the final solution of c_i^*.*

Consider that the right-hand side of Formula (6) contains other variables, $c_j, \forall j \in S\setminus\{i\}$. We need to eliminate the value of c_j and discuss the final solution of P1.

Note that Formula (6) represents a prise-wise function and can be written in three parts:

Part 1. $c_i^* = 0$, if $R \leq \alpha_i$.

Part 2. $c_i^* = \sqrt{R\alpha_i} - \alpha_i$, if $\alpha_i < R \leq \frac{(M_i+\alpha_i)^2}{\alpha_i}$.

Part 3. $c_i^* = M_i$, if $R > \frac{(M_i+\alpha_i)^2}{\alpha_i}$.

c_i^* denotes the optimal strategy of edge node i and only appears in one of the above three parts. We then analyze the optimal solutions of all edge nodes $\mathbf{c}^*$ and obtain the following propositions. We start by discussing whether there exists c_i^* in Part 1.

Proposition 1. $\forall i \in S, c_i^* \neq 0$.

Proof. Firstly, we consider that the optimal solutions of all edge nodes appear in Part 1, i.e., $\forall i \in S, c_i^* = 0$. We will prove that this situation is infeasible using the contradiction method. Assume that this situation is feasible, i.e., $\forall i \in S, c_i^* = 0$. We obtain $\alpha_i = \sum_{j\in S\setminus\{i\}} c_j^* = 0$. Then, we have $R \leq \alpha_i = 0$, which is contradicted with the condition $R > 0$. Thus, there must exist c_i^* that is not equal to zero.

Secondly, we consider the situation that the optimal solutions of some edge nodes appear in Part 1 while those of others appear in Part 2, i.e., $\exists i, j \in S, c_i^* = 0, c_j^* = \sqrt{R\alpha_j} - \alpha_j$. If this proposition exists, it follows that $R \leq \alpha_i$ and $\alpha_j < R \leq \frac{(M_j+\alpha_j)^2}{\alpha_j}$. Note that $R \leq \alpha_i = \sum_{k\in S} c_k = c_j^* + \alpha_j = \sqrt{R\alpha_j}$. Hence, we conclude that $R \leq \alpha_j$. This is contradicted with $R > \alpha_j$. Thus, this situation does not exist.

Thirdly, we consider the situation where the optimal solutions of some edge nodes belong to Part 1 while others belong to Part 3, i.e., $\exists i, j \in S, c_i^* = 0, c_j^* = M_j$. Similarly, we have $R \leq \alpha_i$ and $R > \frac{(M_j+\alpha_j)^2}{\alpha_j} = \frac{(\sum_{k\in S} c_k)^2}{\alpha_j} > \sum_{k\in U} c_k > \alpha_i$. The two inequalities are contradictory; hence, this situation does not exist either.

According to the above results, Proposition 1 is proved. □

Remark 2. *We can infer from Proposition 1 that no c_i^* appears in Part 1. This indicates that all of the edge nodes are willing to accommodate the offloaded cloud task. These edge nodes will achieve revenue as long as they cooperate with the cloud node.*

Then, we discuss the optimal solutions for edge nodes that belong to Parts 2 and 3, and propose Proposition 2.

Proposition 2. *Sort $M_i, i \in [1, n]$ in descending order. Let $m \in [1, n-1]$ denote the number of c_i^* that resides in Part 2. Thus, the number of c_i^* residing in Part 3 is $(n-m)$. The optimal solution of edge node i is*

$$c_i^* = \begin{cases} \frac{Rm-2mA-R+\sqrt{R^2(m-1)^2+4RmA}}{2m^2}, & i = 1, \ldots, m, \\ M_i, & i = m+1, \ldots, n, \end{cases} \tag{18}$$

where $A = \sum_{j=m+1}^{n} M_j$, if the reward R satisfies

$$\frac{(mM_{m+1}+A)^2}{(m-1)M_{m+1}+A} \leq R \leq \frac{(mM_m+A)^2}{(m-1)M_m+A}. \tag{19}$$

Proof. We first sort c_i^* in a specific order such that the first few $c_i^*, i \in [1, m]$ reside in Part 2 and the last few $c_i^*, i \in [m+1, n]$ reside in Part 3. Then, we have

$$c_i^* = \begin{cases} \sqrt{R\alpha_i} - \alpha_i, & i = 1, \ldots, m, \\ M_i, & i = m+1, \ldots, n. \end{cases} \tag{20}$$

We can derive from Formula (20) that $c_i^* = \sqrt{R\alpha_i} - \alpha_i, \forall i \in [1, m]$. It follows that $c_i^* + \alpha_i = \sum_{j=1}^{n} c_j^* = \sqrt{R\alpha_i}$. Thus, we can conclude that $c_1^* = c_2^* = \ldots = c_m^*$. After substituting the conclusion into $\sum_{j=1}^{n} c_j^* = \sqrt{R\alpha_i}, \forall i \in [1, m]$, we have $mc_i^* + A = \sqrt{R((m-1)c_i^* + A)}$, where $A = \sum_{j=m+1}^{n} M_j$. Then, we obtain $c_i^* = \frac{Rm-2mA-R+\sqrt{R^2(m-1)^2+4RmA}}{2m^2}, \forall i \in [1, m]$. The complete expression of c_i^* is shown in Formula (18).

Next, we discuss the value range of the reward R offered by the cloud node, when each edge node takes the optimal cost $\mathbf{c}^*$.

After substituting Formula (18) into $c_i^* \leq M_i, \forall i \in [1, m]$, we have $\frac{Rm-2mA-R+\sqrt{R^2(m-1)^2+4RmA}}{2m^2} \leq M_i$. Then, we obtain $R \leq \frac{(mM_i+A)^2}{(m-1)M_i+A}, \forall i \in [1, m]$, i.e., $R \leq min[\frac{(mM_i+A)^2}{(m-1)M_i+A}]$. If we sort $M_i, i \in [1, m]$ in descending order, we have $R \leq \frac{(mM_m+A)^2}{(m-1)M_m+A}$.

Meanwhile, we have

$$c_i^* > c_j^*, \tag{21}$$

where $i \in [1, m]$, and $j \in [m+1, n]$. For $i \in [1, m]$, we have $c_i^* = \sqrt{R\alpha_i} - \alpha_i$, where $\alpha_i < R \leq \frac{(M_i+\alpha_i)^2}{\alpha_i}$. It follows that $c_i^* + \alpha_i = \sqrt{R\alpha_i}$. For $j \in [m+1, n]$, we have $c_j^* = M_j$, where $R > \frac{(M_j+\alpha_j)^2}{\alpha_j} = \frac{R\alpha_i}{\alpha_j}$. It follows that $\alpha_i < \alpha_j$, so we obtain $c_i^* > c_j^*$, i.e., the optimal solution of an edge node residing in Part 2 is larger than that of an edge node residing in Part 3.

We can derive from Formula (21) that $\frac{Rm-2mA-R+\sqrt{R^2(m-1)^2+4RmA}}{2m^2} > c_j$. Then, we have $R \geq \frac{(A+mM_j)^2}{A+(m-1)M_j}, \forall j \in [m+1,n]$, i.e., $R \geq max[\frac{(A+mM_j)^2}{A+(m-1)M_j}]$. Similarly, it follows that $R \geq \frac{(A+mM_{m+1})^2}{A+(m-1)M_{m+1}}$, if $M_i, i \in [m+1,n]$, are sorted in descending order.

If we sort $M_i, i \in [1,n]$ in descending order, we can conclude Formula (19). Thus, Proposition 2 is proved. □

Finally, we discuss the optimal solutions for those edge nodes residing only in Part 2 or only in Part 3, respectively. The results are reported as Proposition 3.

Proposition 3. *If the optimal solutions of all edge nodes reside in Part 2, i.e., $m = n$, the optimal value of edge node i is*

$$c_i^* = \frac{(n-1)R}{n^2}, 0 < R \leq \frac{n^2 M_n}{n-1}. \tag{22}$$

If the optimal solutions of all of edge nodes reside in Part 3, i.e., $m = 0$, the optimal value of edge node i is

$$c_i^* = M_i, R > \frac{(\sum_{j=1}^{n} M_j)^2}{\sum_{i=2}^{n} M_i}. \tag{23}$$

Proof. This proof consists of two parts.

The first part is to find the results, where the optimal solution of all edge nodes reside in Part 2. We can derive from Formula (6) that $\forall i \in S, c_i^* = \sqrt{R\alpha_i} - \alpha_i$. It follows that $c_i^* + \alpha_i = \sum_{j \in S} c_j^* = \sqrt{R\alpha_i}$. Then, we have $c_1^* = c_2^* = \cdots = c_n^*$. Thus, the result is $c_i^* = \frac{(n-1)R}{n^2}$. We can derive from $c_i^* \leq M_i$ that the constraint of the reward is $R \leq min_{i \in S}(\frac{n^2 M_i}{n-1}) = \frac{n^2 M_n}{n-1}$.

The second part is to find the results, where the optimal solution of all edge nodes resides in Part 3. It follows that $c_i^* = M_i, R > \frac{(M_i+\alpha_i)^2}{\alpha_i}$. Then, we can derive from $R > \frac{(M_i+\alpha_i)^2}{\alpha_i}$ that $R > max_{i \in S} \frac{(\sum_{j \in S} M_j)^2}{\sum_{i \in S \setminus \{i\}} M_i} = \frac{(\sum_{j=1}^{n} M_j)^2}{\sum_{i=2}^{n} M_i}$.

So far, we can conclude that Proposition 3 is proved. □

After combining the aforementioned three propositions, we can conclude the final results of the optimal resource strategy $\mathbf{c}^*$ as three cases.

Case 1: $c_i^* = \frac{(n-1)R}{n^2}, 0 < R \leq \frac{n^2 M_n}{n-1}$.

Case 2: $c_i^* = \begin{cases} N_i, & i = 1, \ldots, m, \\ M_i, & i = m+1, \ldots, n, \end{cases}$

$\frac{(mM_{m+1}+A)^2}{(m-1)M_{m+1}+A} < R \leq \frac{(mM_m+A)^2}{(m-1)M_m+A}$, where $N_i = \frac{Rm-2mA-R+\sqrt{R^2(m-1)^2+4RmA}}{2m^2}$, $A = \sum_{j=m+1}^{n} M_j$.

Case 3: $c_i^* = M_i, R > \frac{(\sum_{j=1}^{n} M_j)^2}{\sum_{i=2}^{n} M_i}$.

Remark 3. *We can observe from Propositions 2 and 3 that $\mathbf{c}^*$ is only in Part 2 and Part 3, or is a combination of Part 2 and 3. We reconsider $\mathbf{c}^*$ under the following three cases to derive the final optimal solution of P1.*

We name the three cases as the insufficient case, the transition case, and the sufficient case. In the insufficient case (Case 1), the reward R cannot attract any edge node to provide its maximum resource for hosting the offloaded cloud task. All the edge nodes will contribute an equal amount of resources based on the number of the participated edge nodes n. In the transition case (Case 2), the amount of resources provided by edge nodes depends on M_i. Those edge nodes with small M_i will provide their maximum resources, while edge nodes with large

M_i will allocate partial resources, which are still more than the maximum resources of the former kind of edge nodes. In the sufficient case (Case 3), inspired by sufficient R, all edge nodes will provide their maximum resources to achieve a large profit.

Figure 2 depicts the optimal resource strategy for edge node i. For any given R, we can determine the intermediate variable m based on M_i. From m, we obtain the corresponding strategy results of c_i^*. We can observe from Figure 2 that m displays a negative correlation with R. With higher rewards from a cloud task, all involved edge nodes tend to provide more resources to win more revenue. Thus, the resource from more edge nodes will reach the resource cost budget. We can also observe that c_i^* displays a positive correlation with R. This indicates that the cloud node can control the number of resources offered by edge node i by adjusting its reward R.

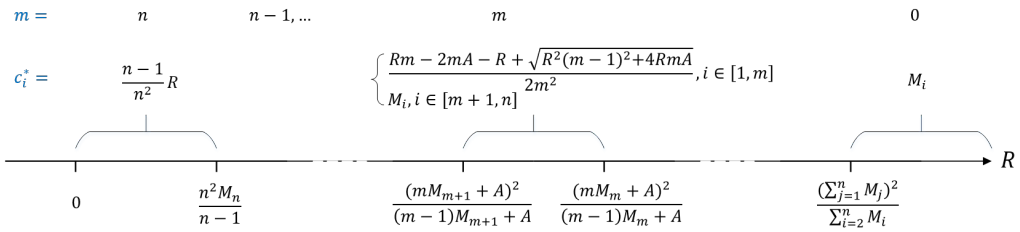


Figure 2. Optimal resource strategy.

4.3. Optimal Reward Strategy

According to the above results of the Resource Determination game, we can further solve the Pricing Reward game. To derive the optimal reward strategy for a cloud node, we compute the derivative of the utility function u_0 to R under three cases.

In Case 1, substituting Formula (22) into the utility function Formula (2) of a cloud node with reward R, we have $u_0 = \gamma\frac{n-1}{n}R - R$. Then, we obtain the derivative of u_0 with respect to R as follows:

$$\frac{\partial u_0}{\partial R} = \gamma\frac{n-1}{n} - 1, \tag{24}$$

where the weight of the resource cost $\gamma > 1$.

The value of Formula (24) is influenced by the variable γ. We analyze the optimal solution of the reward R as follows:

- If $\gamma > \frac{n}{n-1}$, we have $\frac{\partial u_0}{\partial R} > 0$, and $R^* = \frac{n^2 M_n}{n-1}$.
- If $1 < \gamma \leq \frac{n}{n-1}$, we have $\frac{\partial u_0}{\partial R} \leq 0$, and $R^* = 0$. Since $R > 0$, the situation $1 < \gamma \leq \frac{n}{n-1}$ is infeasible.

In Case 2, substituting Formula (23) into Formula (2), we have $u_0 = \gamma(\frac{Rm-2mA-R+\sqrt{R^2(m-1)^2+4RmA}}{2m} + A) - R, m \in [1, n-1]$. The derivative of u_0 with respect to R in Case 2 is

$$\frac{\partial u_0}{\partial R} = \frac{\gamma(m-1)}{2m} + \frac{\gamma(m-1)^2 R + 2mA}{2m\sqrt{R^2(m-1)^2 + 4RmA}} - 1. \tag{25}$$

The value of Formula (25) is also influenced by the variable γ. Thus, it is infeasible to derive a specific value of Formula (25) and give a general solution in Case 2. However, once γ is given, we can determine the value of Formula (25) and obtain the specific solution for Case 2.

To ease the presentation, we set $\gamma = 2$ as an example. Hence, we can infer from Formula (25) that

$$\frac{\partial u_0}{\partial R} = \frac{(m-1)^2 R + 2mA}{m\sqrt{R^2(m-1)^2 + 4RmA}} - \frac{1}{m}. \tag{26}$$

Then, we obtain the results as follows:

- If $m \geq 2$, it follows that $\frac{\partial u_0}{\partial R} > 0$. Then, we obtain $R^* = \frac{(2M_2 + \sum_{i=3}^{n} M_i)^2}{M_2 + \sum_{i=3}^{n} M_i}$, where $m = 2$.
- If $m = 1$, it follows that $\frac{\partial u_0}{\partial R} < 0$. Then, we obtain $R^* < \frac{(M_2 + \sum_{i=2}^{n} M_i)^2}{\sum_{i=2}^{n} M_i}$.

Hence, we obtain $R^* = \frac{(M_2 + \sum_{i=2}^{n} M_i)^2}{\sum_{i=2}^{n} M_i}$, $m = 2$ for Case 2. Note that if $M_1 = M_2 = \ldots = M_l$ or $M_1 > M_2 = \ldots = M_l$ $(l > 2)$, we have $m = l$.

Similarly, in Case 3, the derivative of the utility function u_0 with respect to R is

$$\frac{\partial u_0}{\partial R} = -1 < 0. \tag{27}$$

Thus, we have $R^* < \frac{(\sum_{i=1}^{n} M_i)^2}{\sum_{j=2}^{n} M_j}$.

Combining the results of the above three cases, we find that

$$R^* = \frac{(M_2 + \sum_{i=2}^{n} M_i)^2}{\sum_{i=2}^{n} M_i} \tag{28}$$

is the final optimal solution if we set $\gamma = 2$.

For other settings of the parameter γ, the optimal solution of R can also be obtained in the same way. For a given γ, the optimal reward strategy is related to the resource cost budget of each edge node, M_i. The cloud node tends to announce more reward to those edge nodes with larger M_i, to make them provide more resource costs. This can help the cloud node to win higher utility.

Next, we use Theorem 2 to indicate that the SE is unique when γ is given.

Theorem 2. *The SE* $(R^*, \mathbf{c}^*)$ *for the proposed Stackelberg game is unique for a given* γ.

Proof. As aforementioned, the optimal reward strategy is unique when the resource cost budgets $M_i, \forall i \in S$ and γ are given. The values of these two variables are known and fixed during the entire process of the Stackelberg game. Thus, the optimal reward R^* is unique. Meanwhile, from the results of $\mathbf{c}^*$ in the last subsection, the resource cost each edge node pays is unique under a given R. Thus, we can conclude that the SE for our game is unique under a given γ. □

5. Efficient Algorithm for Solving the Optimal Strategy

In this section, we present an efficient algorithm to decide the optimal strategy for the cloud node and the edge node.

Algorithm 1 illustrates the computation process of the Stackelberg Equilibrium. The input includes the resource cost budget M_i of each edge node. The output includes the optimal strategy of the cloud node R^*, the optimal strategies of edge nodes $\mathbf{c}^*$, and the corresponding utilities u_0 and u_i.

Algorithm 1 aims to calculate the optimal resource strategy of the involved edge node and the optimal reward strategy of the cloud node. The cloud utility and the edge node utility can be calculated via Formulas (1) and (2), respectively. In Algorithm 1, the leader's optimal strategy should be solved first, and then the followers' optimal strategies can be decided accordingly. Hence, we first calculate the value of R^* based on Formula (28) and then calculate the value of

c_i^* according to R^* and M_i (lines 1–2 in Algorithm 1). After calculating the values of R^* and c_i^*, we obtain the cloud utility u_0 and the edge node utility u_i (lines 3–4 in Algorithm 1).

Algorithm 2 shows how to determine c_i^*. The optimal resource strategy c_i^* is influenced by the value of R^*. After solving R^*, we are able to analyze the relationship of R^* and M_i, and determine which case c_i^* resides in. If the value of R^* is small, $m = n$ and c_i^* reside in Case 1 according to Formula (22) (lines 1–2 in Algorithm 2). If the value of R^* is large, $m = 0$ and c_i^* should reside in Case 3 according to Formula (23) (lines 3–4 in Algorithm 2). If the value of R^* is medium, $0 < m < n$ and c_i^* reside in Case 2 (lines 5–8 in Algorithm 2). For the last situation, the value of m needs to be determined first such that we can decide the specific interval of R^* according to Formula (19) (line 6, Algorithm 2). Then, the value of c_i^* can be calculated based on Formula (18) (lines 7–8, Algorithm 2).

Algorithm 1: Computation of the SE.

Require: Sort M_i of all edge nodes in the descent order.

Ensure: R^*, c_i^*, u_0, u_i

1: $R^* \leftarrow \frac{(M_2+\sum_{i=2}^{n} M_i)^2}{\sum_{i=2}^{n} M_i}$

2: $c_i^* \leftarrow Cal\text{-}c_i^*(R^*, M_i), \forall i \in S$

3: $u_0 \leftarrow \gamma \sum_{i \in S} c_i^* - R^*$

4: $u_i \leftarrow \frac{c_i^*}{\sum_{j \in S} c_j^*} R^* - c_i^*, \forall i \in S$

5: **return** R^*, c_i^*, u_0, u_i

Algorithm 2: $Cal\text{-}c_i^*(R^*, M_i)$

Require: R^*, M_i in a descent order

Ensure: c_i^*

1: **if** $0 < R^* \leq \frac{n^2 M_n}{n-1}$ **then**

2: $\quad c_i^* \leftarrow \frac{(n-1)R}{n^2}, \forall i \in S$

3: **else if** $R^* > \frac{(\sum_{j=1}^{n} M_j)^2}{\sum_{i=2}^{n} M_i}$ **then**

4: $\quad c_i^* \leftarrow M_i, \forall i \in S$

5: **else**

6: $\quad m \leftarrow Cal\text{-}m(R^*, M_i)$

7: $\quad c_i^* \leftarrow \frac{Rm-2mA-R+\sqrt{R^2(m-1)^2+4RmA}}{2m^2}, \forall i \in [1, m]$

8: $\quad c_i^* \leftarrow M_i, \forall i \in [m+1, n]$

9: **end if**

10: **return** c_i^*

Algorithm 3 aims to calculate the value of m, the number of c_i^* that resides in Part 2. Through comparing R^* with the combination of M_i (line 3 in Algorithm 3), the value of m can be determined (line 4 in Algorithm 3).

The running time of the whole process of computing the SE is $O(m * (n - m))$. It indicates that the SE can be found in polynomial time with our algorithms.

Algorithm 3: *Cal-m*(R^*, M_i)

Require: R^*, M_i in a descent order
Ensure: m
1: **for** each $i \in S$ **do**
2: $A \leftarrow \sum_{j=m+1}^{n} M_j$
3: **if** $\frac{(mM_{m+1}+A)^2}{(m-1)M_{m+1}+A} \leq R \leq \frac{(mM_m+A)^2}{(m-1)M_m+A}$ **then**
4: $m \leftarrow i$
5: **end if**
6: **end for**
7: **return** m

6. Rethinking Our Incentive Mechanism in Dynamic Edge–Cloud Computing Environment

The above sections report how to find the SE in the edge–cloud computing environment, where all edge nodes are fixed. The edge–cloud computing environment is usually dynamic. New edge nodes may join and existing edge nodes may leave at any time. Once an edge node joins or leaves the edge–cloud computing environment, the calculated SE may change. In fact, the set of edge nodes competing to accommodate that offloaded cloud task directly impacts the optimal strategies for the cloud and each involved edge node. In this section, we investigate how the SE changes along with the joining or leaving of edge nodes and identify the optimal strategies after the change of the edge node set.

We first consider the case that only a single edge node joins or leaves the edge–cloud computing environment. We start by calculating the SE under the setting of n edge nodes and 1 cloud node. We then recalculate the SE when one edge node joins or leaves such that the number of edge nodes becomes $n+1$ or $n-1$. Let $(R^*, \mathbf{c}^*)$ denote the original SE, while $(\tilde{R^*}, \tilde{\mathbf{c}^*})$ denotes the updated SE after an edge node joins or leaves. Meanwhile, we assume that the change of the edge node set is only known by the cloud node. That is, other edge nodes are unaware of such a change.

In the case that edge node i leaves the edge–cloud computing environment, the SE changes only when

$$u_0(\tilde{R^*}, \tilde{\mathbf{c}^*}) \geq u_0(R^*, \mathbf{c}^*) - u_0(R^*, c_i). \tag{29}$$

Here, $u_0(\tilde{R^*}, \tilde{\mathbf{c}^*})$ and $u_0(R^*, \mathbf{c}^*)$ denote the utility of the cloud node under the new SE and the original SE, respectively. Let $u_0(R^*, c_i)$ denote the contribution of edge node i to the utility of the cloud node under the original SE. Note that the utility of the cloud node grows when one more edge node joins the edge–cloud computing environment; hence, Inequation (29) always holds. Thus, the best way to find the new SE is to re-implement the Stackelberg game with the remaining $n-1$ edge nodes when an edge node leaves.

When a new edge node joins the edge–cloud computing environment, the SE changes only when

$$u_0(\tilde{R^*}, \tilde{\mathbf{c}^*}) \geq u_0(R^*, \mathbf{c}^*). \tag{30}$$

According to Formula (28), Inequality (30) always holds as well. When an edge node joins the network, the amount of edge nodes grows. To earn more profits and maximize the utility, the cloud node has the incentive to increase its reward R. The competition among all edge nodes becomes serious than before. This motivates all the edge nodes to provide more resources for serving the offloaded cloud task. Thus, the utility of the cloud node increases

when one more edge node joins the edge–cloud computing environment. To derive the new SE, the best way is to re-implement the Stackelberg game with $n + 1$ edge nodes.

As for the case that multiple edge nodes join the edge–cloud computing environment, the way to find the new SE is the same as that of adding a single new edge node. When multiple edge nodes join or leave the network, the best strategy to find the new SE is to re-implement the Stackelberg game with the latest set of edge nodes.

7. Performance Evaluation

To evaluate the performance of our incentive mechanism, we implement the incentive mechanism through the following setup. First, in Section 7.2, we measure three performance metrics to evaluate the performance of the proposed incentive mechanism, including the reward and utility of the cloud node, the edge node's utility, and the impact of the resource cost budget on the utilities of the cloud node as well as edge nodes. Then, in Section 7.3, we compare the SE with other mechanisms, including the random assignment, the specific assignment, and the social welfare maximum. In the following Section 7.4, we analyze the system-level utility and resource cost of the proposed mechanism and compare it with the results of the social welfare maximum. In Section 7.5, the comparison with the state-of-the-art mechanism [38] is presented. Finally, in Section 7.6, we analyze the impacts of the dynamic behaviors of edge nodes on the SE. The results show that our incentive mechanism strongly motivates the edge nodes to contribute more to accommodating the offloaded cloud task and has obvious advantages. Our mechanism obtains the most resources from edge nodes while providing the lowest reward. It is also robust to realistic dynamic situations.

7.1. Settings of Evaluation

To present the performance of our incentive mechanism, we simulate the results of the SE under different scenarios. The factors that affect the performance of the SE include the number of edge nodes and the resource cost budget. We vary the number of edge nodes from 100 to 1000 with an increment of 100 to simulate different-scale scenarios. Assume that the variable M_i, i.e., the resource cost budget of edge node i, is randomly sampled from a uniform distribution over $[1, M_{max}]$. We test three values for M_{max}, 20, 60, and 100, to see the impact of the resource cost budget on the SE. That is 30 different scenario settings in total. For each setting, we randomly generate 100 scenarios and the average result is reported. We set γ to 2 as analyzed in Section 4.3.

All the experiments are coded in python and conducted on a PC with 1.80 GHz CPU and 8 GB memory.

7.2. Performance Evaluation Results

We first show the SE of the incentive mechanism proposed in this paper and discuss the impacts of the edge node number n and the resource cost budget M_i.

7.2.1. The Reward and Utility of the Cloud Node

Figure 3a,b depicts the reward and utility of the cloud node at the SE, with respect to the increasing number of edge nodes, respectively. We can observe from Figure 3a that for a given M_{max}, the reward of the SE provided by the cloud node is almost linear to the number of edge nodes. It indicates that along with more edge nodes participating in hosting the offloaded cloud task, the cloud node will provide more reward for them. This gives existing edge nodes a strong incentive to accommodate this task to gain more earnings in turn.

Figure 3b depicts the changing trend of the cloud node utility with respect to the increasing number of edge nodes. This reveals the impact of n on the cloud utility u_0. It is

clear that the cloud utility is almost linear to the number of edge nodes, and has a similar changing trend to the cloud reward. As explained in Section 3.2, the utility (i.e., net profit) of the cloud node is the value of the offloaded cloud task minus the reward it offers to the edge node. When the reward R provided by the cloud node increases, the number of participating edge nodes, the contributed resources, and the produced profit increase accordingly. As a result of our incentive mechanism, the utility of the cloud node grows when it provides more rewards to motivate edge nodes. Thus, the cloud node would benefit from our incentive mechanism since edge nodes become willing to host offloaded cloud tasks.

Comparing Figure 3a,b, we can see that the value of the cloud utility is close to that of the cloud reward under the same setting. More edge nodes would motivate the cloud node to adjust its reward R. This further attracts more edge nodes to serve the offloaded cloud task in turn. In general, the cloud node tends to raise its reward since a higher reward can afford more edge node resources to accommodate the offloaded cloud service.

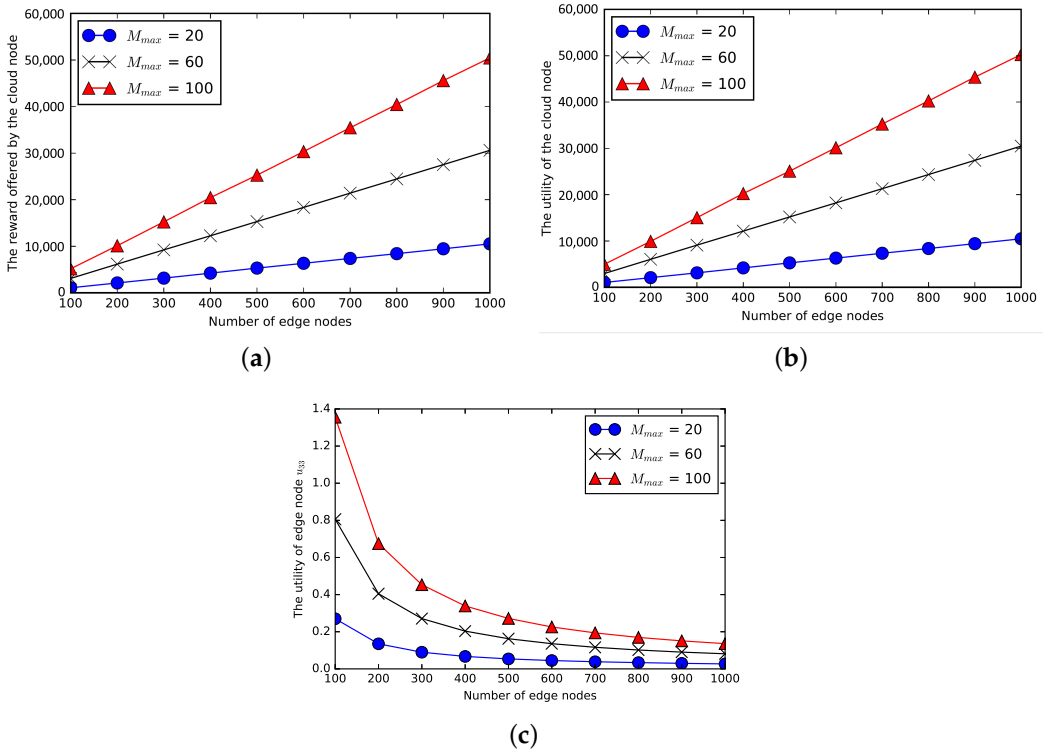


Figure 3. Statistical data of the Stackelberg Equilibrium varying with the number of edge nodes: (**a**) the reward offered by the cloud node; (**b**) the utility of the cloud node; (**c**) the utility of the edge node.

7.2.2. The Resource Cost and Utility of Edge Nodes

We randomly pick a single edge node ($i = 33$) and evaluate its utility. In Figure 3c, it is clear that as new competing edge nodes join, the utility of an existing edge node decreases due to the serious competition among more edge nodes. However, even though the utility of an existing node declines, this edge node is still willing to join because it can obtain payment as long as it hosts the offloaded cloud task. That is, existing edge nodes are willing to provide resources for the offloaded cloud task with our incentive mechanism proposed in this paper.

7.2.3. Impact of the Resource Cost Budget of Edge Nodes on Utilities

Figure 3 plots the impact of M_{max} on the optimal strategies of the cloud node and edge nodes. For any given number of edge nodes, the performance difference between two cases, $M_{max} = 20$ and $M_{max} = 60$, is almost the same as that between two cases, $M_{max} = 60$ and $M_{max} = 100$. This indicates that the resource cost budget of the edge nodes has a serious impact on the utility of the cloud node as well as the utilities of edge nodes. Given the number of edge nodes, these performance metrics are linear to the value of M_{max}. Moreover, with the increasing amount of resource cost budget, the edge nodes can provide more resources to accommodate the offloaded cloud task. Since it has provided more resources to host that cloud service, the cloud node obtains higher utility than the case in which each edge node has a lower resource cost budget. Meanwhile, the high resource cost budget enables each edge node to offer more resources to host that offloaded cloud task and obtain a higher reward from the cloud node in return.

7.3. Comparison with the Benchmarks

This section illustrates the comparison of the proposed incentive mechanism with three benchmarks. The number of edge nodes in the comparison experiments varies from 200 to 1000 with an increment of 200 and M_{max} is set as 50. We set the three benchmarks as follows.

- Random assignment (Ran.). In the random assignment, the cloud node assigns the offloaded cloud task to edge nodes randomly. The resource cost c_i edge node i pays for the offloaded cloud task is random but cannot exceed its cost budget M_i. The total reward the cloud node provides R is random as well.
- Proportional assignment (Prop.). In the proportional assignment, each edge node is assigned a $\frac{M_i}{\sum_{i=1}^{n} M_i}$ portion of the offloaded cloud task. The total reward the cloud node provides is $R = \sum_{i=1}^{n} M_i$.
- Social welfare maximization (SWM). In the social welfare maximization scheme, the objective is to maximize the system-level utility $U_s = u_0 + \sum_{i=1}^{n} u_i$. The respective utility of the cloud node or the edge node is not considered.

Figure 4 presents the performance comparison between the SE of our incentive mechanism and the three benchmarks, including the Ran. scheme, the Prop. scheme, and the SWM scheme. The utilities of the cloud node and edge nodes are shown in Figure 4a and Figure 4b, respectively. The reward offered by the cloud node and the resource cost of the edge node are shown in Figure 4c and Figure 4d, respectively. Note that the resource cost c_i represents the contribution of edge node i to the offloaded cloud task as illustrated in Section 3.2. Since each edge node has the same unit resource cost, the more resource cost the edge node pays, the more it contributes to the task.

First, in Figure 4a, we see that the Prop. scheme, the SWM scheme, and the SE in our mechanism achieve a similar optimal utility for the cloud node for all sizes, while the Ran. scheme achieves low values. For the edge node utility shown in Figure 4b, only the SE achieves a positive value for edge node utility. The edge node utility achieved by the Prop. scheme and the SWM scheme is 0, and that illustrated by the Ran. scheme is negative. In these three schemes, no edge node is willing to host the offloaded cloud task since it cannot receive any net profit from this task. Accordingly, the cloud node motivates no edge node to participate in the offloaded cloud task, so the task cannot be accomplished. Hence, the SE of our incentive mechanism outperforms the three benchmarks (the Ran. scheme, the Prop. scheme, and the SWM scheme).

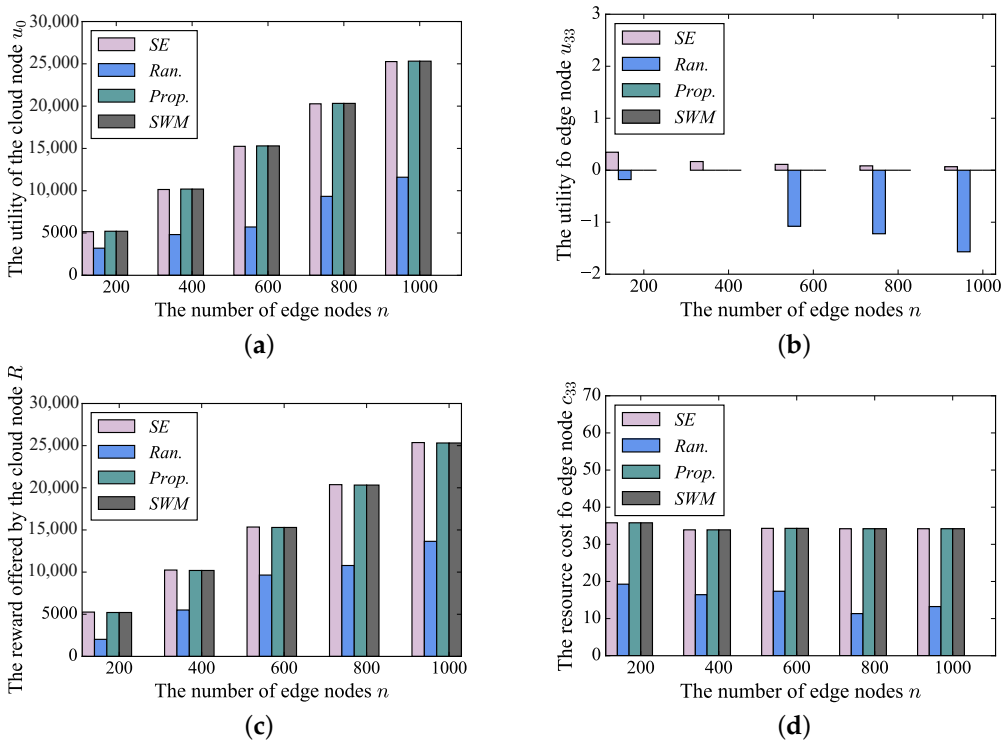


Figure 4. Comparisons between the SE and the benchmarks: (**a**) the utility of the cloud node; (**b**) the utility of the edge node; (**c**) the reward offered by the cloud node; (**d**) the resource cost of the edge code.

Figure 4c,d show the detailed performance of the proposed incentive mechanism and the benchmarks. The performance information includes the reward R from the cloud node and the resource cost c_i of edge node i. We see that the Prop. scheme, the SWM scheme, and the SE of the proposed mechanism illustrate similar rewards offered by the cloud node and the resource cost of edge nodes, but achieve different utilities for edge nodes. This further proves the effectiveness of our mechanism. The Ran. scheme provides less reward and motivates less resource cost contributed by edge nodes, thus achieving lower utility for both the cloud node and edge nodes. Moreover, in Figure 4c, the optimal reward that the cloud node provides for the edge node shares similar trends with the cloud node utility shown in Figure 4a. This indicates that when more edge nodes join the edge–cloud network, providing a higher reward is able to motivate more edge nodes, and more utility will be produced.

From the results, we conclude that the SE of the proposed mechanism produces satisfactory results for both cloud node and edge nodes. Other benchmarks have fatal defects for practical applications. The Prop. scheme and the SWM scheme achieve good result for the cloud node but do not produce any net profit for edge nodes. Thus, edge nodes will quit the task in reality, which will greatly hurt the positivity of the cloud node. The Ran. scheme is irrational and achieves low utility for the cloud node and negative utility for edge nodes. Thus, we believe the SE of the proposed incentive mechanism is an effective and convincing result for the edge–cloud environment.

7.4. Comparison on the System-Level Utility with SWM Scheme

In this section, we present the system-level results achieved by the SE of the proposed incentive mechanism. By definition, SWM indicates the optimal system-level utility (i.e., the total utility of the cloud node and edge nodes) for a scenario. Therefore, the result of the

SWM scheme is presented for comparison. The number of edge nodes in the comparison experiments varies from 200 to 1000 with an increment of 200 and M_{max} is set as 50.

Figure 5 shows the system-level utility U_s (i.e., the social welfare) and the system-level resource cost (i.e., the sum of the resource cost contributed by all edge nodes) in the SE of our incentive mechanism and the SWM scheme. In Figure 5a,b, we see that the SE of our incentive mechanism achieves almost the same system-level utility and contributes the same system-level resource cost to the offloaded cloud task with the SWM scheme. It indicates that our incentive mechanism does not hurt the utility of the whole system when motivating edge nodes to participate in the task. Additionally, we can see that the proposed incentive mechanism earns higher utility (i.e., net profit) for each edge node than the SWM scheme when paying the same system-level resource cost as illustrated in Figure 4b.

We conclude from the results that our incentive mechanism realizes the optimal system-level utility as the SWM scheme when maximizing the utility both of the cloud node and edge nodes. It not only satisfies the rationality and selfishness of edge nodes and motivates edge nodes to participate in the offloaded cloud task, but also realizes the optimal social welfare.

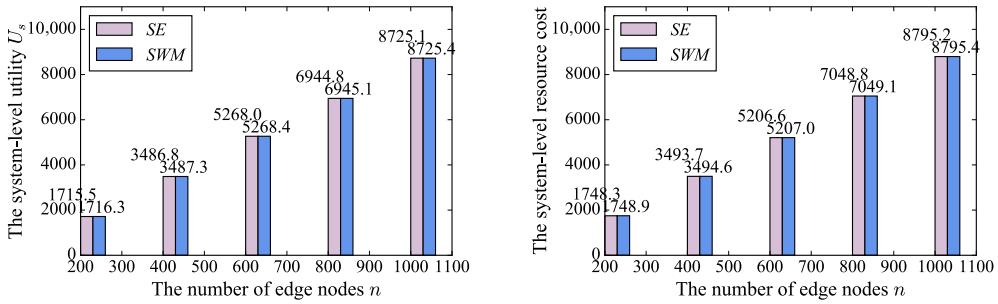


Figure 5. Comparisons between the SE and the SWM scheme: (**a**) the system-level utility; (**b**) the system-level resource cost.

7.5. Comparison with the State-of-the-Art Mechanism

In this section, we compare the SE of our incentive mechanism with the state-of-the-art mechanism. The number of edge nodes in the comparison experiments varies from 200 to 1000 with an increment of 200 and M_{max} is set as 50.

As illustrated in Section 2, the related work about the incentive mechanism in edge computing only focused on the competition between the cloud node and the edge nodes [37,38]. None of them considered the internal competition among edge nodes. The state-of-the-art mechanism was proposed by [38]. It aimed at maximizing the cost reduction for the cloud node and edge nodes and also used the Stackelberg game to design an incentive mechanism for computation offloading. However, it ignored the internal competition among edge nodes, and the reward each edge node received lacked the correlation based on the internal competition. Additionally, the work in [38] did not propose a closed-form optimal solution for its mechanism. The general framework of its algorithm adopted the dichotomy method to search for the optimal reward. For each reward, the algorithm calculated the optimal resource cost of edge nodes and the corresponding utility of the cloud node according to the current value of the reward. After searching, the reward value which illustrated the optimal cloud node utility was selected.

Figure 6a,b demonstrates the utility for the cloud node and edge nodes achieved by the SE of the proposed mechanism and the state-of-the-art mechanism. We can conclude that our result outperforms the state-of-the-art one in the aspect of the utility of the cloud node. However, our algorithm achieves a low value for edge node utility. This is because

the two mechanisms suggest similar resource costs for edge nodes (shown in Figure 6e), but the state-of-the-art mechanism suggests a much higher reward for the cloud node (shown in Figure 6d). This will hurt the positivity of the cloud node greatly. Achieving such a high utility for edge nodes (approximately half the resource cost) is impossible in reality because the cloud node will not provide such a high reward.

We can also see in Figure 6c that our mechanism achieves slightly higher social welfare than the state-of-the-art mechanism. This result indicates a better deployment of the resources of our mechanism. Another advantage of our work is that the running time of our algorithm is very fast—less than 5.6×10^{-3} s for the largest scenario of 1000 edge nodes, as shown in Figure 6f—while the state-of-the-art mechanism is slow and its running time grows exponentially with the problem scale. Thus, our algorithm is more suitable for a realistic edge–cloud environment that contains multiple edge nodes and can handle real-time decision-making.

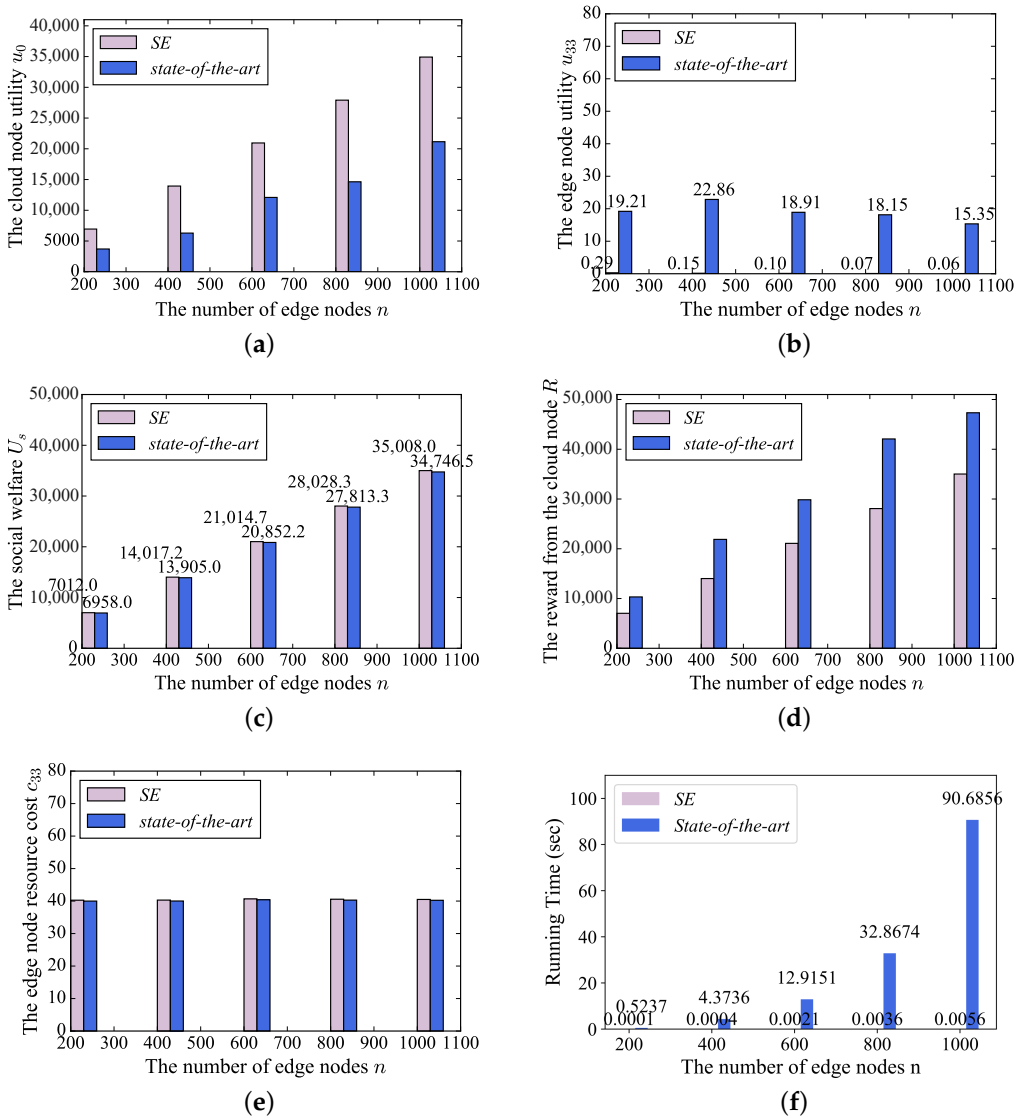


Figure 6. Comparison between the SE of our incentive mechanism and the state-of-the-art mechanism: (**a**) the utility of the cloud node; (**b**) the utility of the edge node; (**c**) the system-level utility; (**d**) the reward offered by the cloud node; (**e**) the utility of the cloud node; (**f**) the utility of the edge node.

7.6. Performance Analysis in the Dynamic Scenarios

In this section, we analyze how the SE changes when edge nodes join or leave the edge–cloud environment. For the sake of clarity in analysis, we construct four types of heterogeneous edge nodes, whose resource cost budget M_i is 200, 150, 100, and 50, respectively. We set 100 edge nodes for each edge node type (i.e., 400 nodes in total). We assume four stages to test the impact of nodes joining, and in each stage, one type of node joins. Accordingly, we design four stages to test the impact of nodes leaving. Nodes join the offloaded cloud task following the order of their resource cost budget. The nodes with the largest cost budget will join at the first stage and nodes with the second-largest cost budget will join at the second stage. The order of leaving is the reverse. That is, the nodes with the smallest cost budget will leave first.

The SE of the dynamic edge–cloud environment is shown in Figure 7. Figure 7c presents the dynamics of the reward R. As new edge nodes join to share the tasks, the cloud node will have to provide more rewards to motivate them (shown in Figure 7c). As a result, we see that the resource cost contributed by edge nodes does not decrease due to the joining of new nodes as shown in Figure 7d. Moreover, as the cloud node claims more and more resources from edge nodes, the utility (net profit) of the cloud node keeps increasing (shown in Figure 7a). Figure 7b shows that the utility of the existing edge nodes decreases due to the arrival of new edge nodes. Their utility is decreasing even if the reward provided by the reward is increasing. But the edge nodes will still provide all their resources as this is still the best choice for them. Additionally, at each SE, the utility of each edge node is related to its resource cost budget. That is, the nodes with the largest resource cost budget will always keep the highest utility.

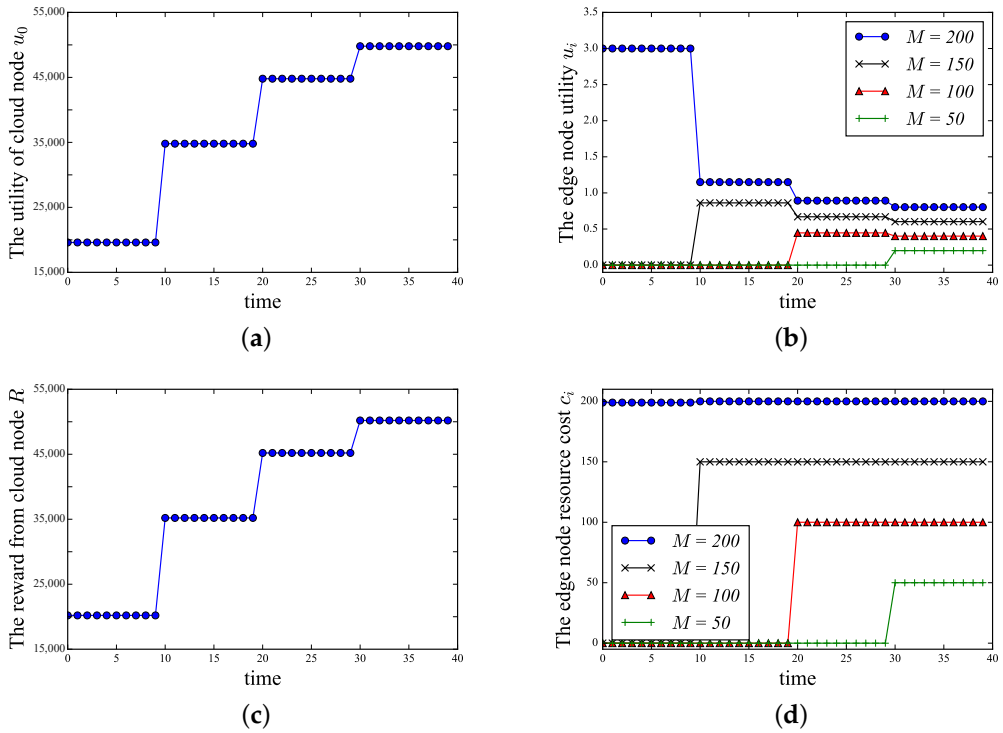


Figure 7. The dynamic of performance metric when edge nodes join the system. (**a**,**b**) present the results of cloud node utility and edge node utility. (**c**,**d**) illustrate the detailed reward R provided by the cloud node and the resource cost c_i contributed by the edge node. The interval of edge nodes joining is 10 s. (**a**) Cloud node utility u_0; (**b**) edge node utility u_i; (**c**) reward R provided by the cloud node; (**d**) resource cost c_i contributed by edge nodes.

Figure 8 shows how the edge node utility changes, which is a mirrored result of edge nodes joining. The resource cost increases when existing edge nodes leave. With wild competition among edge nodes, the reward each participating edge node receives grows. Other metrics of the SE (e.g., the utility of the cloud node) are mirrored to the results of edge nodes joining as well. For simplicity, we do not show the results in this paper.

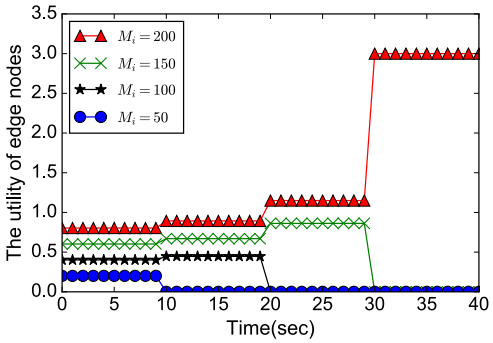


Figure 8. The edge node utility of edge nodes leaving.

The results show that our incentive mechanism is robust for the dynamic edge–cloud environment. The cloud node can adjust its reward accordingly to motivate current participants and claims the highest utility.

8. Discussion

The emerging data-intensive and latency-sensitive applications cause a surge in data processing demands, which challenges network traffic and service latency. Our proposed incentive mechanism enables the cloud node to offload services to edge nodes with permission, thus providing high-quality services to these applications. It motivates edge nodes to contribute their unoccupied resources to host the offloaded cloud task with a reward in return.

One limitation in the practical implementation of our mechanism is that the edge node may cheat when participating in hosting the offloaded cloud service, such as by lying about its provided resource amount to obtain the extra reward. While our current model assumes truthful reporting of resources, future work could incorporate verification mechanisms to ensure honesty. Many works have researched trustworthy, fair, and protected profit transactions using blockchain and smart contracts in related fields, such as private data sharing [43,44] and data trading [45] in edge computing. Integrating these approaches into our incentive mechanism could potentially address this limitation.

Another aspect to consider is the scalability of our approach. While we have demonstrated its efficiency with up to 1000 edge nodes, larger edge–cloud environments might present additional challenges. The polynomial-time algorithm we proposed shows good performance in our experiments, but extremely large-scale deployments may require further optimizations or distributed computation approaches.

Our mechanism currently focuses on the offloading of a single cloud task to edge nodes. In real-world scenarios, multiple cloud tasks from different cloud providers might need to be offloaded simultaneously. This introduces additional complexity in resource allocation and incentive design. In future work, we intend to study the scenario of offloading multiple cloud tasks to edge nodes. Accordingly, we will analyze how to design the incentive mechanism between multiple cloud nodes and a set of edge nodes, potentially incorporating cooperative game theory to model the interactions among cloud providers.

The dynamic nature of edge–cloud environments also presents interesting research directions. While our current approach adapts well to nodes joining or leaving the system,

more complex dynamics such as fluctuating resource availability or varying quality of service requirements could be explored in future work.

9. Conclusions

This paper designs an effective incentive mechanism to motivate edge nodes to accommodate offloaded cloud tasks. We model the incentive problem based on the Stackelberg game, where the cloud node is the leader while the edge nodes are the followers. Beyond the competition between the cloud node and edge nodes, we also impose internal competition among the edge nodes, making our model more realistic than previous approaches.

We proved that the proposed incentive mechanism has a unique Stackelberg Equilibrium and presented the detailed solving process. We then developed an algorithm for computing SE within polynomial time, making our approach practical for real-world deployment. We further analyzed the change in SE if any edge node joins or leaves the edge–cloud computing environment, demonstrating the robustness of our approach. Through extensive evaluation, we compared our incentive mechanism with three benchmarks and the state-of-the-art mechanism. The results indicate that our incentive mechanism can effectively motivate edge nodes to participate in the edge–cloud environment. Our mechanism achieves the most resources from edge nodes while providing the lowest reward compared to alternative approaches. It is also robust to realistic dynamic situations where edge nodes join or leave the environment.

By enabling efficient offloading of cloud services to edge nodes, our work contributes to the development of edge–cloud computing paradigms that can better support data-intensive and latency-sensitive applications. The proposed incentive mechanism balances the interests of both cloud providers and edge node operators, creating a sustainable ecosystem for edge–cloud collaboration.

Author Contributions: Conceptualization, C.Y. and Z.L.; methodology, C.Y.; software, C.Y.; validation, C.Y.; formal analysis, J.X.; investigation, J.X.; resources, J.X.; data curation, C.Y.; writing—original draft preparation, C.Y.; writing—review and editing, J.X.; visualization, C.Y.; supervision, Z.L.; project administration, Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partially supported by the National Natural Science Foundation of China under Grant No.61772544, the National key research and development program under Grant No. 2018YFB1800203, and the National Program for Support of Top-Notch Young Professionals of National Program for Special Support of Eminent Professionals, Tianjin Science and Technology Foundation under Grant No. 18ZXJMTG00290.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Mahmud, R.; Buyya, R. *Fog Computing: A Taxonomy, Survey and Future Directions*; Springer Berlin/Heidelberg, Germany, 2016; pp. 103–139.
2. Mach, P.; Becvar, Z. Mobile Edge Computing: A Survey on Architecture and Computation Offloading. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 1628–1656. [CrossRef]
3. hmed, A.; Ahmed, E. A Survey on Mobile Edge Computing. In Proceedings of the 10th International Conference on Intelligent Systems and Control (ISCO), Vietri sul Mare, Italy, 16–18 May 2016; pp. 1–8.
4. Bonomi, F.; Milito, R.A.; Zhu, J.; Addepalli, S. Fog computing and its role in the internet of things. In Proceedings of the Military Communications and Information Systems Conference (MCC), Oeiras, Portugal, 8–9 October 2012; pp. 13–16.
5. Mohan, N.; Kangasharju, J. Edge-Fog cloud: A distributed cloud for Internet of Things computations. In Proceedings of the Cloudification of the Internet of Things (CIoT), Paris, France, 23–25 November 2016; pp. 1–6.

6. Yang, B.; Chai, W.K.; Xu, Z.; Katsaros, K.V.; Pavlou, G. Cost-Efficient NFV-Enabled Mobile Edge-Cloud for Low Latency Mobile Applications. *IEEE Trans. Netw. Serv. Manag.* **2018**, *15*, 475–488. [CrossRef]
7. Wang, S.; Urgaonkar, R.; Zafer, M.; He, T.; Chan, K.S.; Leung, K.K. Dynamic service migration in mobile edge-clouds. In Proceedings of the 2015 12th Working IEEE/IFIP Conference on Software Architecture, Washington, DC, USA, 4–8 May 2015; pp. 1–9.
8. He, T.; Khamfroush, H.; Wang, S.; Porta, T.L.; Stein, S. It's Hard to Share: Joint Service Placement and Request Scheduling in Edge Clouds with Sharable and Non-Sharable Resources. In Proceedings of the 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), Vienna, Austria, 2–5 July 2018; pp. 365–375.
9. Pasteris, S.; Wang, S.; Herbster, M.; He, T. Service Placement with Provable Guarantees in Heterogeneous Edge Computing Systems. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 514–522.
10. Poularakis, K.; Llorca, J.; Tulino, A.M.; Taylor, I.; Tassiulas, L. Joint Service Placement and Request Routing in Multi-cell Mobile Edge Computing Networks. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 10–18.
11. Chun, B.G.; Ihm, S.; Maniatis, P.; Naik, M.; Patti, A. Clonecloud: Elastic execution between mobile device and cloud. In Proceedings of the Sixth Conference on Computer Systems, Salzburg, Austria, 10 April 2011; ACM: New York, NY, USA, 2011; pp. 301–314.
12. Yang, L.; Cao, J.; Cheng, H.; Ji, Y. Multi-user computation partitioning for latency sensitive mobile cloud applications. *IEEE Trans. Comput.* **2014**, *64*, 2253–2266. [CrossRef]
13. Farhadi, V.; Mehmeti, F.; He, T.; Porta, T.L.; Khamfroush, H.; Wang, S.; Chan, K.S. Service Placement and Request Scheduling for Data-intensive Applications in Edge Clouds. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 1279–1287.
14. Josilo, S.; Dán, G. Wireless and Computing Resource Allocation for Selfish Computation Offloading in Edge Computing. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 2467–2475.
15. Zhang, W.; Li, S.; Liu, L.; Jia, Z.; Zhang, Y.; Raychaudhuri, D. Hetero-Edge: Orchestration of Real-time Vision Applications on Heterogeneous Edge Clouds. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 1270–1278.
16. Ouyang, T.; Li, R.; Chen, X.; Zhou, Z.; Tang, X. Adaptive User-managed Service Placement for Mobile Edge Computing: An Online Learning Approach. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 1468–1476.
17. Mohan, N.; Zavodovski, A.; Zhou, P.; Kangasharju, J. Anveshak: Placing Edge Servers In The Wild. In Proceedings of the 2018 Workshop on Mobile Edge Communications, SIGCOMM, Budapest, Hungary, 20 August 2018; pp. 7–12.
18. Silva, P.; Pérez, C.; Desprez, F. Efficient Heuristics for Placing Large-Scale Distributed Applications on Multiple Clouds. In Proceedings of the 2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), Cartagena, Colombia, 16–19 May 2016; pp. 483–492.
19. Sahoo, J.; Salahuddin, M.A.; Glitho, R.H.; Elbiaze, H.; Ajib, W. A Survey on Replica Server Placement Algorithms for Content Delivery Networks. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 1002–1026. [CrossRef]
20. Huang, Y.; Song, X.; Ye, F.; Yang, Y.; Li, X. Fair Caching Algorithms for Peer Data Sharing in Pervasive Edge Computing Environments. In Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 5–8 June 2017; pp. 605–614.
21. Bernardini, C.; Silverston, T.; Festor, O. MPC: Popularity-based caching strategy for content centric networks. In Proceedings of the 2013 IEEE International Conference on Communications (ICC), Budapest, Hungary, 9–13 June 2013; pp. 3619–3623.
22. Xie, J.; Qian, C.; Guo, D.; Wang, M.; Shi, S.; Chen, H. Efficient Indexing Mechanism for Unstructured Data Sharing Systems in Edge Computing. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 820–828.
23. Zhang, H.; Zeng, K. Pairwise Markov Chain: A Task Scheduling Strategy for Privacy-Preserving SIFT on Edge. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 1432–1440.
24. Myerson, R.B. *Game Theory*; Harvard University Press: Cambridge, MA, USA, 2013.
25. Fernando, N.; Loke, S.W.; Rahayu, W. Mobile cloud computing: A survey. *Future Gener. Comput. Syst.* **2013**, *29*, 84–106. [CrossRef]
26. Dinh, H.T.; Lee, C.; Niyato, D.; Wang, P. A survey of mobile cloud computing: Architecture, applications, and approaches. *Wirel. Commun. Mob. Comput.* **2013**, *13*, 1587–1611. [CrossRef]
27. Chen, X. Decentralized computation offloading game for mobile cloud computing. *IEEE Trans. Parallel Distrib. Syst.* **2014**, *26*, 974–983. [CrossRef]

28. Liu, Y.; Lee, M.J.; Zheng, Y. Adaptive multi-resource allocation for cloudlet-based mobile cloud computing system. *IEEE Trans. Mob. Comput.* **2015**, *15*, 2398–2410. [CrossRef]
29. Zhou, B.; Dastjerdi, A.V.; Calheiros, R.N.; Srirama, S.N.; Buyya, R. mCloud: A context-aware offloading framework for heterogeneous mobile cloud. *IEEE Trans. Serv. Comput.* **2015**, *10*, 797–810. [CrossRef]
30. Shih, C.S.; Wang, Y.H.; Chang, N. Multi-tier elastic computation framework for mobile cloud computing. In Proceedings of the 2015 3rd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering, San Francisco, CA, USA, 30 March–3 April 2015; IEEE: Piscataway, NJ, USA; pp. 223–232.
31. Gao, B.; Zhou, Z.; Liu, F.; Xu, F. Winning at the starting line: Joint network selection and service placement for mobile edge computing. In Proceedings of the IEEE INFOCOM 2019-IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1459–1467.
32. Meng, J.; Tan, H.; Xu, C.; Cao, W.; Liu, L.; Li, B. Dedas: Online Task Dispatching and Scheduling with Bandwidth Constraint in Edge Computing. In Proceedings of the IEEE INFOCOM 2019-IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 2287–2295.
33. Chen, S.; Jiao, L.; Wang, L.; Liu, F. An online market mechanism for edge emergency demand response via cloudlet control. In Proceedings of the IEEE INFOCOM 2019-IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 2566–2574.
34. Zhou, Z.; Liao, H.; Gu, B.; Huq, K.M.S.; Mumtaz, S.; Rodriguez, J. Robust mobile crowd sensing: When deep learning meets edge computing. *IEEE Netw.* **2018**, *32*, 54–60. [CrossRef]
35. Shen, F.; Zhang, G.; Zhang, C.; Yang, Y.; Yang, R. An incentive framework for resource sensing in fog computing networks. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–6.
36. Zheng, Z.; Song, L.; Han, Z.; Li, G.Y.; Poor, H.V. A Stackelberg game approach to proactive caching in large-scale mobile edge networks. *IEEE Trans. Wirel. Commun.* **2018**, *17*, 5198–5211. [CrossRef]
37. Zeng, M.; Li, Y.; Zhang, K.; Waqas, M.; Jin, D. Incentive mechanism design for computation offloading in heterogeneous fog computing: A contract-based approach. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–6.
38. Liu, Y.; Xu, C.; Zhan, Y.; Liu, Z.; Guan, J.; Zhang, H. Incentive mechanism for computation offloading using edge computing: A Stackelberg game approach. *Comput. Netw.* **2017**, *129*, 399–409. [CrossRef]
39. Kang, X.; Wu, Y. Incentive Mechanism Design for Heterogeneous Peer-to-Peer Networks: A Stackelberg Game Approach. *arXiv* **2014**, arXiv:1408.0727.
40. Yang, D.; Xue, G.; Fang, X.; Tang, J. Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing. In Proceedings of the 18th Annual International Conference on Mobile Computing and Networking, Istanbul, Turkey, 22–26 August 2012; pp. 173–184.
41. Boyd, S.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press, Inc.: Cambridge, UK, 2004; pp. 215–231.
42. Kang, X.; Zhang, R.; Liang, Y.; Garg, H.K. Optimal Power Allocation Strategies for Fading Cognitive Radio Channels with Primary User Outage Constraint. *IEEE J. Sel. Areas Commun.* **2011**, *29*, 374–383. [CrossRef]
43. Reniers, V.; Van Landuyt, D.; Viviani, P.; Lagaisse, B.; Lombardi, R.; Joosen, W. Analysis of architectural variants for auditable blockchain-based private data sharing. In Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, Limassol, Cyprus, 8–12 April 2019; pp. 346–354.
44. Shi, P.; Wang, H.; Yang, S.; Chen, C.; Yang, W. Blockchain-based trusted data sharing among trusted stakeholders in IoT. *Softw. Pract. Exp.* **2021**, *51*, 2051–2064. [CrossRef]
45. Huang, Y.; Zeng, Y.; Ye, F.; Yang, Y. Fair and protected profit sharing for data trading in pervasive edge computing environments. In Proceedings of the IEEE INFOCOM 2020-IEEE Conference on Computer Communications, Toronto, ON, Canada, 6–9 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1718–1727.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.