Profit Sharing for Data Producer and Intermediate Parties in Data Trading over Pervasive Edge Computing Environments

Yaodong Huang, Yiming Zeng, Fan Ye, and Yuanyuan Yang

Abstract—Innovative edge devices (e.g., smartphones, IoT devices) are becoming much more pervasive in our daily lives. With powerful sensing and computing capabilities, users can generate massive amounts of data. A new business model has emerged where data producers can sell their data to consumers directly to make money. However, how to protect the profit of the data producer from rogue consumers that may resell without authorization remains challenging. In this paper, we propose a smart-contract based protocol to protect the profit of the data producer while allowing consumers to resell the data legitimately. The protocol ensures the revenue is shared with the data producer over authorized reselling, and detects any unauthorized reselling. We also introduce a data relay process that can enhance data accessibility in wireless edge networks. We formulate a revenue sharing problem to maximize the profit of both the data producer and resellers/relayers. We formulate the problem into a two-stage Stackelberg game and determine a ratio to share the reselling revenue between the data producer and resellers/relayers. Extensive simulations show that with resellers and relayers, our mechanism can achieve up to 49.5 percent higher profit for the data producer and resellers/relayers.

Index Terms—Pervasive edge computing, blockchain, smart contract, game theory, profit sharing, data trading

1 Introduction

With the arrival of 5G networking systems, edge computing is becoming increasingly pervasive in our daily lives. The backbone technologies help the thrive of smart edge devices, e.g., IoT devices, phones, and vehicles. These edge devices are equipped with advanced sensing and communicating capabilities and can create massive amounts of data, which can be transferred and shared easily among devices and clients. Some new business models are emerging with an abundance of devices and data. Producers, the owner of certain devices, can provide information or services to consumers for income. One example is "We Media", where data producers trade their contents, mainly video clips or texts, to other consumers and make money.

Consider a situation where the producer has for-profit content to sell to potential consumers in peer edge environments. The producer wants to get reasonable rewards for the data it sells, and the consumer wants to get desired and genuine data from the producer. Most current solutions require a trusted third party or platform to manage content and subscriptions. For example, Gumroad [1] provides this kind of service for producers to sell digital contents directly to consumers. Although considerable amounts of data are sold on these platforms, there are still adverse events [2], mostly related to security, trust, and privacy concerns. Meanwhile, the consumer cannot assure that the purchased data are genuine on such platforms, and unauthorized reselling is regulated largely by user reports, at best incomplete, untimely, and unreliable [3].

To ensure secure and reliable data access, we adapt blockchain technology for data trading in edge environments. The blockchain technology used widely in cryptocurrencies is a secure ledger in distributed environments. The ledger stores transactions between accounts to transfer value, such as currencies, assets, and general data [4], [5]. There are many security features built into the blockchain. First, the completed transaction history is encoded in blocks for restoration and verification. Second, unless malicious users have more than half of the total computational power, neither the block nor its contained data can be modified theoretically. Blockchain technology helps a transaction take place in a decentralized fashion, thus improving efficiency, security, and privacy over a network without a centralized entity or a trusted third party.

Despite the advantages of blockchain technology for data trading in such distributed systems, there are many challenges in the data trading/reselling context. First, the consumer can resell purchased data to other consumers and share the profit with the original data producer. The data producer should know that the data item is resold and the second consumer can verify that the data item is genuine. Second, to achieve fast and reliable data access in edge environments, it is crucial that data items are proactively stored onto some devices. Then, users can get data from nearby devices for quick access. Third, since the resources of devices are often limited in edge environments, any device that stores and sends data needs to consume its resources. Thus, such devices should be compensated appropriately with a share of the revenue from the producer.

• *The authors are with the Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY 11794 USA. E-mail: {yaodong.huang, yiming.zeng, fan.ye, yuanyuan.yang}@stonybrook.edu.*

Manuscript received 19 June 2020; revised 5 Apr. 2021; accepted 8 Apr. 2021. Date of publication 15 Apr. 2021; date of current version 5 Dec. 2022. (Corresponding author: Yaodong Huang.) Digital Object Identifier no. 10.1109/TMC.2021.3073669

In this paper, we study the data trading problem in peer edge environments. We propose a selling and reselling mechanism that ensures proper profit for the data producer, while ensuring data genuineness to the consumer. The trading and revenue sharing between different nodes are protected and enforced by the smart-contract and blockchain. To determine the optimal revenue sharing ratio, a Stackelberg game is formulated to describe the interaction between the producer and reseller/relayer nodes, and a unique equilibrium is derived. We also propose a rounding scheme for the relaxed Stackelberg game model and derive the performance guarantee between the rounded result and the optimal result. Extensive simulations show that our proposed mechanism can achieve up to 49.5 percent higher profit for the producer and also share profit for other nodes to incentivize their participation.

We make the following contributions in this paper.

- We design a profit sharing mechanism for devices to resell data and share the revenue with the data producer. We develop a smart contract-based protocol to ensure that the data selling and reselling are trackable and the profit of each party is publicly accepted.
- We design a protocol for the consumer to verify if the data is genuine from the producer without imposing extra burdens on the consumer and protect the profit of both the consumer and the producer.
- We model the interaction between the producer and resellers/relayers as a two-stage Stackelberg game to determine the optimal revenue sharing ratio. We further derive the analytical solutions to maximize the revenue of both the producer and resellers/relayers.
- We analyze the data storage and delivery scheme and prove that the approximation ratio between the rounded equilibrium of the Stackelberg game and the optimal results is 2.
- We implement our proposed mechanism and conduct extensive evaluations. The results show that the proposed mechanism can achieve up to 49.5 percent more profit compared with using only the producer for data selling, while offering profit for resellers and relayers to incentivize participation in the network.

The rest of this paper is organized as follows. In Section 2 we discuss the model of the profit sharing mechanism. In Section 3 we propose and analyze our Stackelberg game for profit sharing. In Section 4 we evaluate the proposed data trading mechanism. In Section 5 we discuss some related work on peer edge networks and smart contracts. Finally, we conclude the paper in Section 6.

2 Reselling and Relaying Processes

In this section, we introduce the data trading model in edge environments. We discuss the reselling and relaying processes in the model. We then present the details for the processes under authorized and unauthorized reselling situations and the detection of unauthorized reselling. We also discuss the relaying process in detail.

2.1 Overview

2.1.1 Roles of Nodes

There are four main roles for each node $n \in \mathcal{N}$, which indicates an active device in the system. The *producer* is the node that produces original data items. The *consumer* is the node that demands and purchases data items, denoted as $j \in \mathcal{N}$. The *reseller*, denoted as $i \in \mathcal{N}$, purchases and stores data items for self-implementation and can deliver data items to requesting consumers to get more income. The *relayer* determines which data items to be stored and delivers such cached data items to requesting consumers. The relayer is denoted as $i' \in \mathcal{N}$.

2.1.2 Reselling Process

Nodes can take different roles at different times with different data items in the network. In our proposed system, a consumer is allowed to resell the data item after purchasing it. The node that resells the data changes its role from a consumer to a reseller. Since the data item is revealed to the reseller after purchasing, it can sell data in one way or another, authorized or pirated. Thus, the goal of our design is to make sure the producer is aware of this reselling transaction and get corresponding revenue from the transaction.

We assume that consumers and the producer are honest and rational when a reselling transaction happens. The consumer wants to make sure that the data item it receives is genuine and not corrupted. Thus, the consumer will proactively check if the data is from the data source. The selling information is encoded in the blockchain. Once it receives data item k, consumer j will present verification packets to potential producers for checking. Meanwhile, the producer wants to ensure it gets a share of the reselling revenue. Since all selling information is stored in the blockchain, the producer can trace which nodes have already purchased the data item. Once the producer gets the packet from a new consumer that matches the information of a data item it once produced, it will check if the reselling is authorized. If the reselling is authorized, which means the reseller informs the producer about the selling and revenue sharing information, the transaction will be completed. The revenue is shared between reseller i and the producer at a previously determined ratio r_{ik} for data item k. Otherwise, the producer shows proof of data ownership and rejects the transaction.

2.1.3 Relaying Process

In wireless edge environments, proactive caching is a crucial mechanism to share data among peer devices [6]. Placing data onto nodes closer to potential consumers improves the availability, reduces the latency for data retrieval, and enhances the experience of consumers.

Relayers are nodes that have available caching capacities and want to make a profit by using them. The producer wants to improve the consumer experience by putting data items in multiple places closer to consumers. Thus, the producer offers rewards to motivate relayers to cache data items proactively. The relayers need to pay a fee to the producer initially to store the data items. To minimize chances of unauthorized copying, relayers store encrypted data so they cannot view the content directly. The producer will also share the revenue with relayers for each data item delivered to consumers. Once a consumer requests and purchases a data item that could be served by the relayer directly, the relayer will send the encrypted data item to the

consumer, and the consumer will get the key for decrypting the data item.

2.1.4 Assumptions

We assume that all nodes are selfish, rational, and want to make more profit in the data selling process. Most of the nodes follow the rules to get a reasonable share of their profit. There are a small number of rogue nodes that want to have unfair advantages to get more profit by cheating. For example, in a reselling process, producers may want to have all revenue without sharing it with the reseller; the reseller may want to have all revenue without notifying the producer.

In our proposed mechanism, the participation of consumers is the key to reselling and ensures more profit. We assume that consumers will check whether data items are genuine through the validation process, either willingly or mandatorily. Detecting situations that the consumer intends to buy pirated data items and conduct off-network transactions is beyond the scope of this paper.

We now present an authorized reselling process and an unauthorized process, and how the unauthorized process can be detected and dealt with.

2.2 An Authorized Reselling Process

In an authorized reselling process, the reseller will inform the producer about the reselling process and share the revenue with the producer. The revenue sharing is ensured using smart contracts. Smart contract [7] is a protocol that once it is signed by involved parties, the contract content can be viewed and validated by others. The smart contract is then encoded in blocks and then can be enforced by users in the blockchain.

When a consumer finds data items that it demands, it then sends requests to a nearby reseller to get the data item. The corresponding reseller then sends the data item and conducts key exchanges with the consumer. Meanwhile, the reseller will generate a tripartite contract, which involves the producer, the reseller, and the consumer. The contract has the data item information, selling price, and revenue sharing ratio between the reseller and the producer. The price of this data item is determined by the producer while the revenue sharing ratio is previously determined. Then, the consumer pays for the data and three parties sign the contract using their respective private keys. The contract now takes effect and is stored into the blockchain to be publicly validated. Fig. 1 shows the reselling process.

Since the contract takes effect only when the consumer, producer, and reseller sign, it can avoid situations where one or more parties do not play with the preset rules. If the contract has incorrect information such as false price or false sharing ratio, the consumer or the producer can simply deny signing it. If the producer wants to have all the revenue, it has to deny the contract and make a new contract. However, the consumer and reseller will not sign this new contract if the original information in the previous contract is correct. Thus, the producer will not get any revenue. Meanwhile, since each signature can be verified through the public keys of these three parties, the contact effectiveness

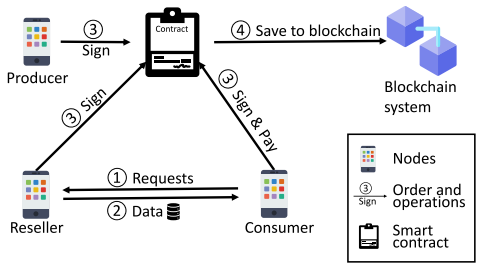


Fig. 1. Step by step information for an authorized reselling process. (1) The consumer requests a data item from a nearby reseller. (2) The reseller then sends the data item to the consumer. (3) The reseller generates a smart contract. The producer, consumer, and reseller all need to sign the contract to make it effective. (4) The contract is encoded in the blocks and enforced.

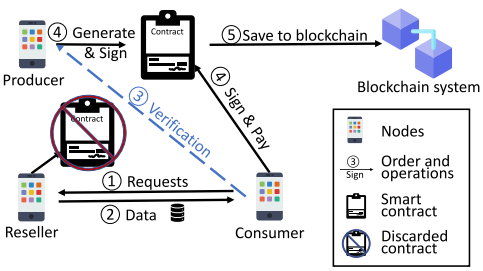


Fig. 2. Step by step information for an unauthorized reselling process. (1) The consumer requests a data item from a nearby reseller. (2) The reseller then sends the data item to the consumer. (3) The consumer broadcast a piece of verification information and the data producer will notice. (4) The data producer generates a smart contract directly with the consumer, and the contact from the reseller will be discarded. (5) The contract is encoded in the blocks and enforced.

can be verified. Other nodes then acknowledge the credit change of each node.

Note that this reselling process will not corrupt the user privacy in the blockchain. The revenue sharing is similar to three-party transactions in traditional cryptocurrencies, and the data content will not be revealed.

2.3 An Unauthorized Reselling Process

In an unauthorized reselling process, the reseller does not inform the producer about the reselling process. The reseller pretends the data item as of its own to take all revenue alone. The resellers will rewrap the data item and tell others this is a new data item.

Once a consumer finds the data item that it demands, it may send the requests to this rogue reseller, which pretends itself as the producer of the data item. The rogue reseller sends the data and conducts key exchanges. Here, the consumer checks whether the data item is genuine. The consumer works on the verification process and broadcasts its verification information. Once the real producer receives the verification information, it can easily check whether there exists a data item it once produced. If the producer detects the data item and it is resold without authorization, the producer will present the information to prove it. The detailed validation process is presented in Section 2.4. Then the producer will generate a new smart-contact in which it leaves the reseller out. The consumer and the producer will sign the new contract, and all the revenue is given to the producer. Fig. 2 shows the process of unauthorized reselling.

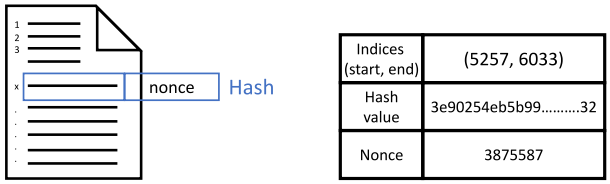


Fig. 3. The hashed item (a) and the verification packet example (b). The hash included a part from the data and a nonce to meet the previous set pattern of the hash value.

Since the producer will get notified by the consumer about an unauthorized resold data item, the smart contract from the reseller will not be signed by the other two parties. In this situation, the reseller will suffer a loss for delivering the data to the consumer. Thus, the reseller cannot get unfair advantages by masquerading.

2.4 Verification Process

As we mentioned above, the consumer checks whether the data item is genuine. When the consumer receives a data item, it needs to check if the corresponding data producer is not masqueraded by conducting the verification process.

Similar to the mining process, the consumer generates a hash from the data item that is hard to obtain but easy to check. The consumer first chooses any part of the data from the data item it receives, and hash together with a nonce, shown in Fig. 3a. It continues to use different nonce until getting a hash value that follows a certain pattern (like several 0's in front of the hash value). The stricter the pattern is, the harder the process will be. This hardness is related to the price of the data item decided by the producer. This process takes some time and computation power, which can help the message against counterfeit due to computational difficulty and keep the data content concealed. Then, the consumer broadcasts the verification packet, shown in Fig. 3b, including which part of the data indices, the hash, and the nonce, to the network. The potential real producer will receive the verification packet and checks if there are data items that match the information. The producer will hash the same content by the indices and nonce to see if it matches the hash value given in the verification packet. If a producer finds out the data item related to the verification packet, it can declare that the transaction is unauthorized. It has to show the consumer 1) it indeed owns the data item, 2) the reseller has purchased the data item before, and 3) the smart contract does not include the producer. Such information can be retrieved from the blockchain and can be easily verified. Then, the producer will generate a new smart contract and the consumer will sign this new contract. The contract by the rogue reseller will be discarded, and the reseller will not receive any revenue. Note that rouge consumers may deny payment for the data item. This problem can also happen when consumers send false information to trusted third parties. A potential solution is to have a reputation system where the consumer will suffer a reputation loss for such behaviors. We will study such solutions in the future.

The verification process does not waste the computational power of the consumer. During the verification process, the consumer contributes "work" to check the data item information by computing a hash following preset hash patterns. In blockchain systems, any valid "work" can be related to credit. In our design, the "work" of the consumer corresponds to some credits, which can be used to pay for the data item. If the verification process does not find any masquerade, the credits are used as the payment to the producer. If the process finds that the reselling is unauthorized, the real producer will raise a new smart contract and the consumer pays using the credits generated. Either way, the "work" of the consumer is not wasted, and we can keep the profit of the producer. Note that resource-limited devices can purchase the computational power from resourceful devices to conduct the verification. The node can hand over a hash and let another node do the verification process, which is still verifiable by the producer.

TABLE 1
Comparison of Resellers and Relayers

Roles	Reseller	Relayer
Who can become?	Previous consumer	Any node
Initial payment to the producer	In full	In part
Data cached	Decrypted data	Encrypted data
Can conduct authorized reselling/relaying?	Yes	Yes
Can conduct unauthorized reselling/relaying?	Yes (have raw data)	No (need key to decrypt data)

2.5 Relaying Process

The design of the relayer is to increase the availability and the security of data items. On one hand, all nodes are profit-seeking and rational. Since nodes have storage spaces, they can make a profit from the available storage by caching and helping nearby consumers to get the data item. On the other hand, caching can make the content easy to obtain for consumers. In edge scenarios, caching can make more replicas for the data item to make it more robust. Caching data onto key locations where they can be requested frequently reduces the overhead of the network [6]. Similar to resellers, nodes that want to become relayers will send the data item to nearby consumers to get revenue. However, there are three key differences between resellers and relayers. First of all, any node can become a relayer once the relaying process can make a profit, while only previous consumers can become a reseller. Second, the initial payment is different. Resellers have already paid in full for the data item, while relayers only need to pay a part. Third, since a reseller is a previous consumer, and has paid the data in full, the data content is revealed to them. Thus, resellers can conduct unauthorized reselling. Relayers are not previous consumers, and the only purpose is to deliver data to make a profit. Thus, the producer wants cached data items encrypted to keep data concealed. Table 1 shows the comparison of these two roles.

In the data relaying process, relayers will negotiate the revenue sharing ratio from the producer, and evaluate the potential revenue of this data item they can get and the cost it needs to pay. Then the relayer will decide whether to cache the encrypted data item. The data item is encrypted

TABLE 2
Notations Used in the Formulation

$i \in \mathcal{I}$	Reseller nodes and reseller node set
$i' \in \mathcal{I}'$	Relayer nodes and relayer node set
$j \in \mathcal{J}$	Consumer nodes and consumer node set
$n \in \mathcal{N}$	Nodes and node set
$k \in \mathcal{K}$	Data items and data item set
r_{ik}	The share of revenue for node i to resell data item k
c_{ijk}	The cost for node i send data item k to node j
p_k	The price of data item k
g_k	The number of resellers for item k
ρ_{jk}	The request of data item k from consumer j
l_{ik}	The cost measurement weight for node i to store data item k
o_k	The size of data item k
x_{ik}	Determination variable for node i to store data item k
y_{ijk}	Determination variable for node i to send data item k to node j
$S(r, y)$	The profit function for the data producer
$U(r, x, y)$	The profit function for the reseller
$U'(r, x, y)$	The profit function for the relayer

using a symmetric cryptographic method (e.g., triple DES, AES), and relayers cannot reveal the content. After relayers caching the corresponding data items, consumers that demand the data item can get it from nearby relayers. Consumers also need to conduct key exchanges with the producer to decrypt the data item. The producer will send the encryption/decryption key to the consumer using an asymmetric cryptography algorithm. Then, the relayer will generate a tripartite contract. The producer, relayer, and consumer will sign the contract, and the transaction is finished. We argue that since the relayer caches the data item on behalf of the producer, and the producer can be impersonated (i.e., rogue resellers), the consumer needs to pay the data through the verification process to check whether the data item is genuine.

Note that compared with the data items, keys are often much smaller in size (e.g., much less than 1KB). Thus, the consumer will get data from relayers and get the key from the producer. The key exchanges will likely be successful even if the consumer is far away from the producer due to the small size. Thus, we design the consumer to have key exchanges directly with the producer. To better protect the data item, the producer will assign different keys for different data replicas on relayers. The consumer who buys the data item will have a corresponding key of the data item it receives from corresponding relayers.

3 Formulations and Solutions to Revenue Sharing Game

In this section, we first formulate the revenue sharing gaming problem and propose a Stackelberg game to model the interaction between the producer and resellers/relayers. We then provide analytical solutions to the model. The notations used in the formulations are listed in Table 2.

3.1 Revenue Sharing Game Formulation

In the reselling and relaying process, the producer shares the revenue with resellers and relayers at a sharing ratio r_{ik} and $r_{i'k}$. We define the revenue sharing ratio between the producer and resellers/relayers as the ratio of the revenue (i.e., price) the resellers and relayers get for reselling a data item. Once a data item is sold, and the consumer purchase is processed, the revenue will be divided using this ratio. This ratio will be encoded in a smart contract and later encoded in the blocks. Thus, the transaction and revenue distribution can be publicly validated and accepted, and the credit for each party is updated accordingly.

To determine this ratio between the producer and resellers, we formulate the problem as a two-stage Stackelberg game for all parties. By offering a profitable ratio, the producer can encourage more nodes to participate and achieve more profit. For each producer, the profit comes from the shared revenue of resellers and the sale directly to consumers. The formulation is as follows.

$$\max_{r} S(r, y) = \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} (1 - r_{ik}) p_k y_{ijk} + \sum_{i' \in \mathcal{I}'} \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} (1 - r_{i'k}) p_k y_{i'jk} + \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} (p_k - c_{0jk}) y_{0jk} + \sum_{k \in \mathcal{K}} g_k p_k \tag{1}$$

$$\text{s.t.} \sum_{i \in \mathcal{I}} y_{ijk} + \sum_{i' \in \mathcal{I}'} y_{i'jk} + y_{0jk} = \rho_{jk} \ (\forall j \in \mathcal{J}, \forall k \in \mathcal{K}), \tag{2}$$

$$y_{ijk} \in \{0, 1\} \ (\forall i \in \mathcal{I} \cup \mathcal{I}' \cup \{0\}, \forall j \in \mathcal{J}, \forall k \in \mathcal{K}), \tag{3}$$

$$r_{ik} \in [0, 1] \ (\forall i \in \mathcal{I} \cup \mathcal{I}', \forall k \in \mathcal{K}). \tag{4}$$

We denote $S(r, y)$ as the objective function of the producer. The objective function (1) consists of two parts. The first part is the revenue shared with resellers. $(1 - r_{ik})$ denotes the remaining revenue sharing ratio for the producer. p_k is the price of data item k. y_{ijk} is a binary determination variable. $y_{ijk} = 1$ indicates that reseller i will resell and send data item k to node j. $y_{i'jk} = 1$ indicates that relayer i' will deliver data item k to node j. The second part is the revenue if the data item is directly sold to consumers by the producer. c_{ijk} is the cost for node i to deliver data item k to consumer j. In peer edge environments, communication cost is one of the most important costs. We use a weighted communication cost in this situation to indicate the price for the data item to be delivered. $i = 0$ indicates that the node is the data producer. Thus, $(p_k - c_{0jk})$ is the profit of the producer for selling data item k to node j, and y_{0jk} determines whether the producer needs to send the data item to node j. $\sum_{k \in \mathcal{K}} g_k p_k$ denotes the revenue from consumers of the previous round, which will be the resellers for the current round. Constraint (2) indicates that if node j demands data item k, which is denoted as $\rho_{jk} = 1$, there is always a node that will send the data item to it. Meanwhile, since ρ_{jk} is a binary parameter, y_{ijk} and $y_{i'jk}$ are binary determination variables, there will be only one node that will deliver a certain data item to a certain consumer. Constraints (3) and (4) are value ranges of variables.

For reseller i, the profit comes from the reselling revenue deducting the cost it pays. The formulation for a specific

data item k and a reseller i is as follows.

$$\max_{x,y} U_i(r,x,y) = \sum_{j\in\mathcal{J}}(r_{ik}p_k - c_{ijk})y_{ijk} - p_k - l_{ik}\ln\frac{1}{1+o_k-x_{ik}o_k} \quad (5)$$

$$\text{s.t.} x_{ik}\rho_{jk} \geq y_{ijk},\ (\forall j\in\mathcal{J},\forall k\in\mathcal{K}), \quad (6)$$

$$x_{ik}, y_{ijk} \in \{0,1\}\ (\forall j\in\mathcal{J},\forall k\in\mathcal{K}). \quad (7)$$

The objective Function (5) consists of three parts. The first part is the profit for this node i which resells data item k. It will get the corresponding share of the revenue r_{ik} if it delivers the data to consumer j. x_{ik} and y_{ijk} are determination variables, where $y_{ijk}=1$ indicates that the node i will deliver data item k to node j, and $x_{ik}=1$ means node i will store data item k. The second part indicates that resellers have paid p_k to the data from the producer or another reseller to get the data. The third part is the cost of storage. Since each data item is of different sizes, the storage impact on different data items is not the same. Thus, inspired by [8], we denote the cost of node i to store k as $\ln\frac{1}{1+o_k-x_{ik}o_k}$, where o_k indicates the size of data item k. Adding 1 to the denominator is to prevent the logarithm from becoming infinite. We use l_{ik} as the weight to measure the storage cost. Constraint (6) indicates that node i delivers data item k to node j only if node i stores the data item and node j requests for the data item. Constraint (7) limits the determination variables to binaries.

For relayer i', the formulation is similar to reseller i but has some differences.

$$\max_{x,y} U'_{i'}(r,x,y) = \sum_{j\in\mathcal{J}}(r_{i'k}p_k - q_k - c_{i'jk})y_{i'jk} - l_{i'k}\ln\frac{1}{1+o_k-x_{i'k}o_k} \quad (8)$$

$$\text{s.t.} x_{i'k}\rho_{jk} \geq y_{i'jk}\ (\forall i'\in\mathcal{I}',\forall j\in\mathcal{J},\forall k\in\mathcal{K}), \quad (9)$$

$$x_{i'k}, y_{i'jk} \in \{0,1\}\ (\forall i'\in\mathcal{I}',\forall j\in\mathcal{J},\forall k\in\mathcal{K}). \quad (10)$$

Relayers can store data items and deliver them to potential nodes that demand them. Thus, the data item from the producer can reach more nodes and relayers can also get revenue to satisfy these demands. Meanwhile, relayers are not previous consumers and have not purchased the data item yet. The objective Function (8) consists of two parts. The first part indicates the profit for relayer i' to deliver data. It will share a part of the revenue with the producer and needs to pay an initial fee to the producer for the data item k. Similar to the reseller, the sharing ratio with the producer is $r_{i'k}$. q_k indicates the payment to the producer for data item k. Here, $q_k = \frac{p_k}{\sum_j \rho_{jk}}$, which indicates the average cost for all potentially profitable demands. The second item in the objective function is the cost of the storage the same as the part of resellers. Constraints (9) and (10) are the same as those of resellers. The formulation ensures that if the reselling or relaying of a data item is not profitable for a reseller or relayer, it will not participate. The decision on reselling or relaying a data item to a consumer is made only when it is profitable.

3.2 Stackelberg Game Model

This game model we proposed above is a two-stage Stackelberg game. In a typical Stackelberg game, there is a leader and several followers. The leader offers incentives and followers respond to the incentive proposed and make caching decisions. In this game, the producer is modeled as the leader whereas resellers and relayers are modeled as followers. We assume that all participants in the game are rational and selfish. In Stage I, the strategy of the producer is to present the revenue sharing ratios r_{ik} to the reseller i who sells the data item k, and $r_{i'k}$ to the relayers i' who delivers the data item k. The producer announces that monetary revenue would be awarded if resellers and relayers deliver the data item to consumers. In Stage II, after given incentive revenue from the producer, resellers and relayers need to determine if they want to deliver the data item to gain their own profit. The game is defined as follows.

- *Followers*: resellers and relayers.
- *Leader*: the producer.
- *Strategies*: the producer determines the revenue sharing ratio r and resellers and relayers determine whether to store and deliver data: x and y.
- *Payoff*: maximize the profit for the producer $S(r,y)$, the total profit for resellers $U(x,y)$, and the total profit for relayers $U'(x,y)$

In our game model, the producer offers the incentives to relayers/resellers first, then relayers/resellers respond with caching strategies. After receiving responses from relayers/resellers, the producer updates its incentive policies and then offers them to relayers/resellers. This process repeats until both the producer and relayers/resellers have no intention to change the balance because they are rational and selfish and they will try to make their benefits the most. In our formulation, the solution is the specific case of the Stackelberg game called equilibrium.

Definition 1 Stackelberg Equilibrium. *The outcome $\{r^*,x^*,y^*\}$ of this two-stage Stackelbergame reach the equilibrium if the following conditions are satisfied for the content provider and every reseller $i\in\mathcal{I}$, relayer $i'\in\mathcal{I}'$ at the same time:*

$$S(r^*,y^*) \geq S(r,y^*),(\forall r) \quad (11)$$

$$U_i(x^*,y^*,r^*) \geq U_i(x,y,r^*),(\forall x,y) \quad (12)$$

$$U'_{i'}(x^*,y^*,r^*) \geq U'_{i'}(x,y,r^*),(\forall x,y), \quad (13)$$

where r^*,x^*,y^* are the optimal value for r,x,y respectively.

3.3 Equilibrium Analysis

In this section, we analyze how to derive the equilibrium of the proposed game. This problem is challenging because the revenue sharing ratio for the producer, and the storage determination for resellers and relayers are coupled together.

To analyze the problem, we separate the process of the game into two different stages. In Stage I, the producer decides the revenue sharing ratio offered to relayers to encourage them to store or buy data items and for resellers to resell purchased data items. In return, resellers and relayers determine which data items to be stored and deliver data items to the requesting consumer in Stage II. This game jointly solves the problem of how to determine the revenue sharing ratio, and storage and delivery determinations.

3.3.1 Stage II

We first address the case in Stage II.

The objective for resellers is to maximize their total profit. In this game, resellers and relayers are followers. After observing the action of the leader (the producer offers the incentive ratio), the storage determination of resellers is determined as the response for cooperation. The optimal strategy of the reseller is decided by solving an optimization problem. This problem takes the revenue sharing ratio offered by the producer as the input. More specifically, the problem for each reseller i with respect to data item k is defined as

$$\max_{x,y} U_i(x,y),$$
$$\text{s.t.}(6),(7),$$

and for each relayer i', with respect to data item k, the problem is defined as

$$\max_{x,y} U'_{i'}(x,y),$$
$$\text{s.t.}(9),(10).$$

The objective function of the problem is discrete due to the discrete variable of x_{ik} and $x_{i'k}$. To solve the reseller problem, the discrete storage variable x_{ik} and $x_{i'k}$ are relaxed from {0,1} to [0,1] which are continuous. Hence, profit Functions (5) and (8) are monotone increasing with determination variables y_{ijk} and $y_{i'jk}$. Resellers and relayers will cache contents to maximize the utility and satisfy the requests of consumers. With limited cache size for resellers and relayers and many content requests, they will not not cache unrequested contents. To maximize the profit of the reseller, the optimal solution achieves when (6) and (9) are transferred as the following,

$$x_{ik}\rho_{jk} = y_{ijk}(\forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall k \in \mathcal{K}), \tag{14}$$

$$x_{i'k}\rho_{jk} = y_{i'jk}(\forall i \in \mathcal{I}', \forall j \in \mathcal{J}, \forall k \in \mathcal{K}). \tag{15}$$

Thus, we can replace variables y_{ijk} and $y_{i'jk}$ by x_{ik} and $x_{i'k}$ accordingly. After replacing the determination variable, the profit function of the reseller (5) and the relayer (8) are continuous about the x_{ik} and $x_{i'k}$ respectively. We first calculate the partial maximization over x of the profit function. The first partial derivative functions are derived as follows,

$$\frac{\partial U_i(x,y)}{\partial x_{ik}} = \sum_j r_{ik}p_k\rho_{jk} - \sum_j c_{ijk}\rho_{jk} - \frac{o_k l_{ik}}{1+o_k-x_{ik}o_k}, \tag{16}$$

$$\frac{\partial U'_{i'}(x,y)}{\partial x_{i'k}} = \rho_{jk}\left(\sum_j r_{i'k}p_k - \sum_j c_{i'jk} - q_k\right) - \frac{o_k l_{i'k}}{1+o_k-x_{i'k}o_k}. \tag{17}$$

To get the maximization value, let $\frac{\partial U(x,y)}{\partial x_{ik}} = 0$, when (16) is a concave function, we have

$$x^*_{ik} = -\frac{l_{ik}}{r_{ik}p_k\sum_j \rho_{jk} - \sum_j c_{ijk}\rho_{jk}} + 1 + \frac{1}{o_k}. \tag{18}$$

For (17) let $\frac{\partial U'(x,y)}{\partial x_{i'k}} = 0$, we have

$$x^*_{i'k} = -\frac{l_{i'k}}{r_{i'k}p_k\sum_j \rho_{jk} - \sum_j c_{i'jk}\rho_{jk} - q_k\rho_{jk}} + 1 + \frac{1}{o_k}. \tag{19}$$

Note that after the relaxation, to guarantee the x^* is in [0,1], the following constraints also need to be satisfied,

$$r_{ik}p_k\sum_j \rho_{jk} - \sum_j c_{ijk}\rho_{jk} \le l_{ik}o_{ik}, \tag{20}$$

$$\sum_j \rho_{jk}(r_{i'k}p_k - c_{i'jk} - q_k) \le l_{i'k}o_{i'k}, \tag{21}$$

$$(r_{ik}p_k\sum_j \rho_{jk} - \sum_j c_{ijk}\rho_{jk})(o_k+1) \ge l_{ik}o_{ik}, \tag{22}$$

$$\sum_j \rho_{jk}(r_{i'k}p_k - c_{i'jk} - q_k)(o_k+1) \ge l_{i'k}o_{i'k}, \tag{23}$$

where (20), (21) correspond to the condition $x^*_{ik} \le 1$ and $x^*_{i'k} \le 1$, and (22), (23) correspond to $x^*_{ik} \ge 0$ and $x^*_{i'k} \ge 0$. If these conditions cannot be satisfied, the optimal solution is reached either $x^* = 0$ or $x^* = 1$. To solve this problem, storage variables are relaxed from $\{0,1\}$ to [0,1], the solution derived is not feasible to the original problem. Hence, we propose an algorithm to get discrete $\{0,1\}$. The performance guarantee is discussed in Section 3.4.

After obtaining the revenue sharing ratio, the optimal storage policy for resellers is derived and regarded as the input to get the optimal revenue sharing ratio of the producer to maximize the profit in Stage I.

3.3.2 Stage I

Now we discuss Stage I.

In Stage I, the producer determines the revenue sharing ratio (i.e., incentive) offered to resellers to maximize the revenue. Thus, the producer considers the anticipated strategy of each reseller. By introducing the optimal storage strategy of the reseller and relayer (18), (19) and constraints (20), (21), (22), (23), the problem to maximize the revenue of the producer can be formulated as follows,

$$\max S(r,y), \tag{24}$$

$$s.t.(2),(14),(15),(18)-(23).$$

We use (14), (15) to replace determination variable y_{0jk}, y_{ijk} with x_{ik} and $x_{i'k}$, and use (18), (19) to replace x_{ik} and $x_{i'k}$ with r_{ik} and $r_{i'k}$ respectively under optimal circumstances. After that, the revenue function only contains the variable r_{ik} as shown in the following,

$$S(r) = \sum_i \sum_k \left(l_{ik} \frac{\sum_j c_{ijk}\rho_{jk} - \sum_j c_{0jk}\rho_{jk}}{r_{ik}p_k \sum_j \rho_{jk} - \sum_j c_{ijk}\rho_{jk}} \right) \tag{25}$$

$$+ \sum_{i'} \sum_k \left(l_{i'k} \frac{\sum_j \rho_{jk}(c_{i'jk} + q_k - c_{0jk})}{\sum_j \rho_{jk}(r_{i'k}p_k - c_{i'jk} - q_k)} \right) \tag{26}$$

$$+ \sum_i \sum_k \frac{(o_k+1)(1-r_{ik})p_k \sum_j \rho_{jk}}{o_k} \tag{27}$$

$$+ \sum_{i'} \sum_k \frac{(o_k+1)(1-r_{i'k})p_k \sum_j \rho_{jk}}{o_k} \tag{28}$$

$$+ \sum_i \sum_k l_{ik} + \sum_k g_k p_k + \sum_j \sum_k (p_k - c_{0jk})\rho_{jk} \tag{29}$$

$$- \sum_i \sum_j \sum_k \frac{(o_k+1)(p_k - c_{0jk})\rho_{jk}}{o_k}. \tag{30}$$

The objective function is not a standard convex optimization problem. The convexity or concavity of the function depends on the parameter setting which corresponds to different solutions. (27) and (28) are linear functions about r_{ik} and $r_{i'k}$, (29) and (30) only contain the constant parameters, and they do not affect the convexity of the revenue function. The convexity of (25) and (26) varies with the parameter setting. We discuss all conditions in detail as follows.

(1) $\sum_j c_{0jk}\rho_{jk} - \sum_j c_{ijk}\rho_{jk} - p_k > 0$ or $\sum_j \rho_{jk}(c_{0jk} - c_{i'jk} - q_k) > 0$.

The revenue function part (25) or (26) is concave about the r_{ik} or $r_{i'k}$, the problem is to maximize the total revenue, the optimal value could be derived from the extreme point of the revenue function. The standard convex optimization techniques [9] can be applied.

(2) $\sum_j c_{0jk}\rho_{jk} - \sum_j c_{ijk}\rho_{jk} - p_k < 0$ or $\sum_j \rho_{jk}(c_{0jk} - c_{i'jk} - q_k) < 0$.

The revenue function part (25) or (26) is a convex function about r_{ik} or $r_{i'k}$. The maximum value is determined by the end point of the r_{ik} or $r_{i'k}$ which is 0 or 1. In this case, the optimal strategy of the reseller or the relayer can be derived by comparing the revenue value of the objective function between 0 and 1, then choose the r_{ik} and $r_{i'k}$ with larger revenue. The optimal solution is as follows,

$$r^*_{ik} = \arg\max_{r_{ik}} \{S(r_{ik} = 0), S(r_{ik} = 1)\}. \tag{31}$$

When $r_{ik} = 1$ or $r_{i'k} = 1$, which means the producer does not need to serve consumers directly, resellers or relayers can satisfy all requests from consumers and it will cost more for the producer to serve consumers directly. When $r_{ik} = 0$ or $r_{i'k} = 0$, the revenue offered to resellers or relayers is larger than the producer serving the requests from consumers directly. Thus, the best strategy of the producer is to serve consumers directly.

(3) $\sum_j c_{0jk}\rho_{jk} - \sum_j c_{ik}\rho_{jk} - p_k = 0$ or $\sum_j \rho_{jk}(c_{0jk} - c_{i'jk} - q_k) = 0$

The revenue function part (25) or (26) is monotone decreasing about the revenue sharing ratio r_{ik} or $r_{i'k}$. Hence, the optimal solution is $r^* = 0$. As mentioned above, the producer can get the largest income when $r_{ik} = 0$ or $r_{i'k} = 0$, which means the optimal strategy for the producer is to sell the data item by itself instead of offering the incentives to resellers.

Note that the convexity or concavity of (25) and (26) is individually determined for each i, i' and k. The final problem is the summation of concave parts decided by the condition above. Since non-concave parts can obtain the optimality directly, we only need to solve concave parts to get the optimal results of the entire problem.

3.4 Performance Analysis

In our proposed game, storage variables x_{ik}, $x_{i'k}$, y_{ijk}, and $y_{i'jk}$ are integers. Directly solving the problem to get optimal results as integers is difficult since it is a mixed non-linear binary programming problem. Previous work [10], [11] shows that this kind of problem is NP-Hard. Thus, in Stage II, storage variables are relaxed from $\{0, 1\}$ to [0,1]. The solution derived after the relaxation may not be feasible to the original problem if it is not integral. To address this problem, we propose the cost-based rounding algorithm and prove that it has an approximation ratio to the optimal result. Since the relayer set and the reseller set have no intersection in (25) to (30), for simplicity, we use i to represent both i (reseller) and i' (relayer) in the remainder of this section. We also denote the result of optimization problem (5) as $U^*(x)$ with the optimal integer result, and $U(x)$ with the relaxed (i.e., relaxed $x_{ik} \in [0, 1]$) one. $U^\dagger(x)$ is the result after rounding (i.e., integer $x^\dagger_{ik} \in \{0, 1\}$). The main idea is to get a sequence of nodes such that the overall cost of all nodes is the largest but not exceeds the optimum value with relaxation. We achieve this by selecting the largest d_{ik} every time until the summation of the selected d_{ik} is larger than the optimal cost with relaxation. We then set the corresponding $x_{ik} = 1$. The detailed information of the rounding algorithm is illustrated in Algorithm 1, where $OPTM_k$ is defined as $\sum_i d_{ik}x_{ik}$ for each k, and $OPTM'_k$ denotes the remaining value of $OPTM_k$ after the deduction in each round.

The objective Functions (5) and (8) contain the same component $-l_{ik} \ln \frac{1}{1+o_k - x_{ik}o_k}$ which is not linear. We relax this component by introducing two linear functions to constrain the logarithmic function in a small, compact region. Fig. 4 illustrates the relationship between the logarithmic function and two linear functions.

In Fig. 4, each linear function is tangent to the logarithmic function and only has one intersection with the logarithmic function at integer x-coordinates. The coordinates of two intersection are $(0, ln(1 + o_k))$ and (1,0). These two linear functions are two extreme conditions to limit the

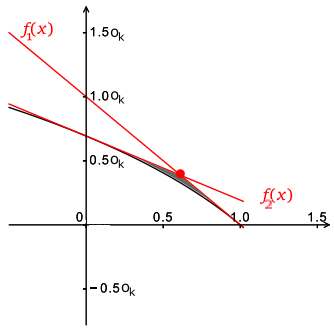


Fig. 4. The relaxation of the non-linear part of the objective function. $f_1(x)$ and $f_2(x)$ are two tangent lines which intersect the non-linear function at integer points.

logarithmic function in the feasible region ($x \in [0,1]$), and guarantee that the logarithmic function is less than or equal to these two linear functions. Two linear functions are represented as follows,

$$f_1(x) = -o_k x + o_k, \tag{32}$$

$$f_2(x) = -\frac{o_k}{1+o_k}x + log(1+o_k). \tag{33}$$

The intersection of these two linear function is at $x_l = \frac{o_k - ln(1+o_k)}{o_k - \frac{o_k}{1+o_k}}$.

After relaxing the logarithmic function into linear functions, we can construct a linear equation $V_i(x)$. The $V_i(x)$ consists of two parts. For any linear component in $U_i(x)$, denoted here as $U_i^{lnr}(x)$, $V_i(x)$ is the same, for the logarithm part in $U_i(x)$, denoted as $U_i^{log}(x)$, $V(x)$ use the aforementioned $f_1(x)$ and $f_2(x)$ in the form. Thus, $V(x) = U_i^{lnr}(x) + f_1(x > x_l) + f_2(x < x_l)$, which is a linear equation. We represent the linear form as $V_i(x) = d_{ik}x_{ik} + c$, where $U_i(x) \leqslant V_i(x)$ within the range $x \in [0,1]$, and d_{ik} is regarded as the summation of all parameters. Hence, the total revenue function can be written as $V(x) = \sum_i d_{ik}x_{ik} + c$. d_{ik} is determined by the optimal result x_{ik}. If $x_{ik} < x_l$, the corresponding d_{ik} and c will use $f_1(x)$; otherwise, d_{ik} and c will use $f_2(x)$. This process creates a segmented unbounded binary linear programming problem with determined d_{ik}.

Algorithm 1. Cost Ronding

Input: Solution of (5) and (8) with $r_{ik}, r_{i'k}$, $D_k = \{d_{ik}\}$.
Output: Rounded determination variable $\{x_{ik}^\dagger\}$.

for All $k \in \mathcal{K}$ **do**
  $OPTM_k = \sum_i d_{ik}x_{ik}$;
  while $OPTM_k > 0$ **do**
    $\mu = \text{argmax } D_k$;
    $x_{\mu k}^\dagger = 1$;^{i}
    $OPTM_k = OPTM_k - d_{\mu k}$;
    $D_k = D_k \backslash d_{\mu k}$
  end while
  $OPTM'_k = OPTM_k$;
  for All do $\{i | d_{ik} \in D_{ik}\}$ **do**
    $x_{ik}^\dagger = 0$;
  end for
end for

Theorem 1. *The cost rounding algorithm (Algorithm 1) is an approximation algorithm to the original problem without rounding and it achieves an approximation ratio of 2, i.e., $U^*(x) \leqslant 2U^\dagger(x)$, under the condition that $OPTM_k > \min_i D_k$ for each k.*

Proof. We prove the theorem by contradiction. For the rounding process, we denote $d_{\mu k}$ as the minimum among all of the d_{ik} associated with those $x_{\mu k}^\dagger$ is rounded to 1. It is clear that $OPTM'_k > 0$, thus $d_{\mu k} \leqslant OPTM_k - OPTM'_k$, where $OPTM'_k$ equals to the reminder of $OPTM_k$ defined in line 9. We assume that $OPTM'_k > \frac{OPTM_k}{2}$. Then, for any $x_{\eta k}^\dagger$ that is not rounded to 1 with related $d_{\eta k}$, it must satisfy that $OPTM'_k - d_{\eta k} < 0$. Otherwise, $x_{\eta k}^\dagger$ will be rounded to 1. Thus, $d_{\eta k} > OPTM'_k > \frac{OPTM_k}{2}$. Meanwhile, since $d_{\mu k}$ is the minimum with associated $x_{\mu k}^\dagger$ rounded to 1, $d_{\mu k} > d_{\eta k}$.

As mentioned above, $d_{\mu k} \leqslant OPTM_k - OPTM'_k$, and as our assumption, $OPTM'_k > \frac{OPTM_k}{2}$, we get $d_{\mu k} < \frac{OPTM_k}{2}$. Since $d_{\eta k} > \frac{OPTM_k}{2}$, we can get $d_{\eta k} > d_{\mu k}$. However, this contradicts the assumption that $x_{\mu k}^\dagger$ is rounded to 1, which indicates $d_{\mu k} > d_{\eta k}$. Thus, we conclude that $OPTM'_k \leqslant \frac{OPTM_k}{2}$, i.e., $V^\dagger(x) - c = \sum_i d_{ik}x_{ik}^\dagger \geqslant \frac{OPTM_k}{2} = \frac{V(x)-c}{2}$, $V^\dagger(x) \geqslant \frac{V(x)}{2} + \frac{c}{2} \geqslant \frac{V(x)}{2}$. Thus, $V^\dagger(x) \geqslant \frac{V(x)}{2}$.

Note that $V^\dagger(x) = U^\dagger(x)$ when using the same rounding policy with the same x_l as the rounding conditions. After rounding, $U_i^{lnr}(x)$ part is still the same for both functions, and $U_i^{log}(x)$ part is the same for $V^\dagger(x)$ and $U^\dagger(x)$ when $x = 0$ or $x = 1$, as we can get from Fig. 4. Since $V^\dagger(x) = U^\dagger(x)$ and $U^*(x) \leqslant V(x)$, we can get

$$U^*(x) \leqslant V(x) \leqslant 2V(x)^\dagger = 2U^\dagger(x).$$

Thus, the approximation ratio can be obtained. □

4 Performance Evaluation

In this section, we evaluate the performance of our proposed mechanism on data selling and reselling. We focus on evaluating the performance of profit shared between the data producer and resellers/relayers over different settings of the network, and whether the reselling makes more profit for nodes. Profit functions are defined in (1), (5) and (8) respectively for the producer, resellers and relayers.

In the simulation, we assume that nodes are distributed randomly in a square area with a density of one node per $25m^2$. We assume every two nodes can directly communicate with each other in the network. The cost for data delivery c_{ijk} is set as the distance between two nodes. We test different strategies for data delivery to evaluate the proposed reselling model. Because the $\sum_k g_k p_k$ is a constant factor, we do not include this profit for the producers in the simulation results. "Producer only" indicates that the producer sends all data items to consumers. Resellers are not involved, and all revenue is going to the producer. "With reseller" indicates that if the consumer is close to a reseller, the reseller will send the data item to it. Otherwise, the consumer will still obtain the data from the producer. The revenue of the data item sold by resellers is shared with the producer. "With reseller & relayer" indicates that both

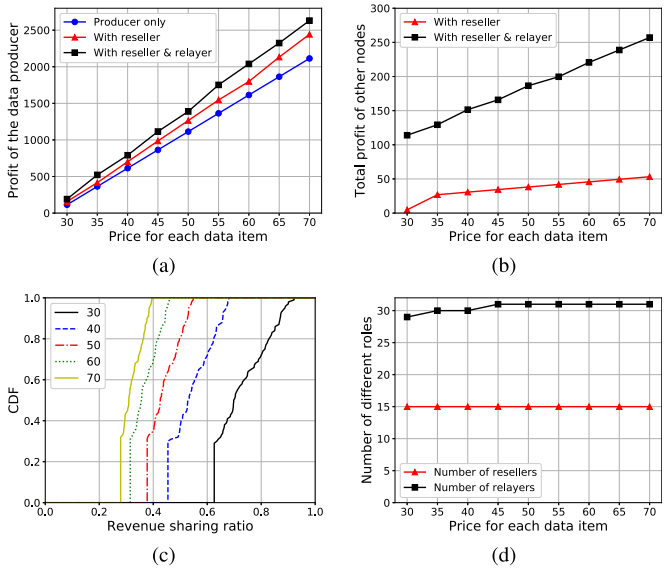


Fig. 5. The profit of the producer (a) and resellers (b) under different prices of data items. The profit increases almost linearly as prices grow. The distribution of revenue sharing ratio under different prices is shown in (c), each line indicates different prices of the data item. The number of resellers and relayers is shown in (d). The higher the price, the less revenue sharing ratio that nodes will require when participating.

resellers and relayers can deliver data to the consumer, and the revenue will be shared with the producer. Nodes can still get data from the producer if the distance is short. Note that every node can be a relayer if the revenue sharing ratio $r_{i'k}$ is reasonable and profitable. For simplicity, we only consider selling one data item over the network. Selling multiple data items can be achieved by applying the mechanism multiple times for each data item.

The convex optimization problem (25)-(30) is implemented and solved using CVXPY [12]. We conduct our simulations on a computer with an Intel Core i7-5820K processor and 32GB RAM. For a fair comparison, we only change one parameter at a time. We use 5 different random seeds for simulations, i.e., the randomness for each simulation is the same for different parameters. Each result shown below is the average of 5 simulations with the same setting.

4.1 Profit on Different Prices of Data Items

We first evaluate the profit that the data producer and resellers/relayers can get under different prices of data items. We set 100 nodes in the area. Among them, 15 nodes are previous consumers that can serve as resellers at this moment. 50 more consumers are demanding the data item, and each demand needs to be satisfied by the producer, a reseller, or a relayer.

Fig. 5 shows the profit of the producer (a) and other nodes (b) over different prices of the data item. In general, the higher the price, the higher the profit nodes will obtain. The profit of the data producer grows linearly with the price if the producer is responsible to satisfy all the demands and deliver corresponding data items. Since the demands are fixed and the communication cost remains constant in a specific parameter setting, the profit grows perfectly linearly with the increasing price of the data item. When resellers participate in delivering items, they can satisfy some demands with lower communication costs for delivering data items. In this simulation set, having resellers will reduce 12.6 percent of communication costs for delivering data. The total communication cost drops from 693 to 606 when using resellers. It can reduce the cost of delivering data to requesting nodes, and the total cost of the network is lower. The revenue of relayers also grows almost linearly with the price. Note that nodes need enough revenue to compensate for their cost if they deliver the data item. Lower prices (e.g., 30 in such situations) will not incentive many resellers to deliver data. With resellers and relayers getting a corresponding share with the delivery, the producer will also get more profit. The producer will have more profit if we introduce relayers into the networks. If a node can make a profit, it may participate and become a relayer to deliver the data item to nearby nodes. Overall, the producer receives 14.2 percent more profit with the help of resellers and 27.3 percent more profit with the help of both resellers and relayers.

Fig. 5c shows the distribution of different revenue sharing ratio $r_{i'k}$ which relayer i' gets when it resells the data item to a consumer. It shows the minimum revenue sharing ratio that can incentivize a certain fraction of nodes to join as relayers because they gain more revenue than the cost for data delivery and data payment. As the price is higher, the revenue sharing ratio is lower for relayers. For instance, when the price of the data item is 50, about 40 percent of the participating nodes require a revenue sharing ratio less than 0.4. When the price increases to 60, about 70 percent of participating nodes will require the same or smaller ratio of the overall price. Since all nodes are selfish and rational, they will need compensation for their cost for data delivery if they participate. Thus, higher prices per data item compensate for the cost of nodes more easily. Relayers can still get profit when they require a lower share of the revenue. This in turn gives more profit to the producer.

Fig. 5d shows the number of resellers and relayers in the network. The number of relayers almost stays the same under different data prices. When the price is higher, a smaller revenue sharing ratio is required to compensate for the cost. For the same number of relayers, this indicates an almost linear increase for the profit as in Fig. 5b. For any data item price, the higher profit shows that our proposed mechanism indeed helps increase the profit for the data producer, while resellers and relayers can also get profit in the process.

4.2 Profit on Different Sizes of Networks

Next, we evaluate the profit of the data producer and resellers/relayers under different sizes of networks. We set 25 to 175 nodes in areas with the same density of nodes. To make sure the same density, there are 4 to 28 nodes which are previous consumers now serving as resellers. The price of the data item is 50, and there are randomly 50 percent nodes requesting the data item.

Fig. 6 shows the profit for the data producer (a) and other nodes (b). In general, the profit of the producer increases as more nodes are in the network. Since we set the same request rate for each network, more nodes bring more requests, and the producer can make more profit by selling more data items. Meanwhile, the profit of resellers and

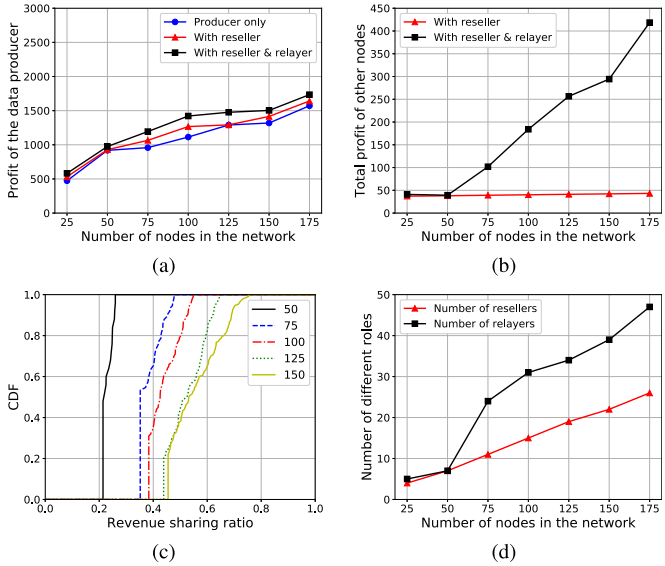


Fig. 6. The profit of the producer (a) and resellers (b) under different sizes of the network. The profit for the producer grows as networks with increasing nodes. If fewer nodes are in the network, the revenue is not shared with resellers. The distribution of revenue sharing ratio under different sizes of the network is shown in (c). The number of resellers and relayers is shown in (d). The larger the network size is, the more nodes will participate, and the higher sharing ratio resellers will require.

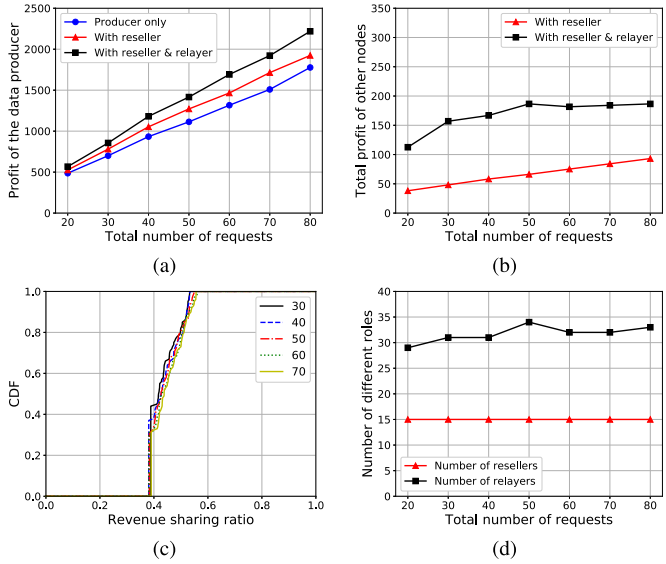


Fig. 7. The profit of producer (a) and resellers (b) under different numbers of requests. The profit for the producer increases as more nodes request. The distribution of revenue sharing ratio under different sizes of the network is shown in (c). The number of resellers and relayers is shown in (d). The participation and revenue sharing ratios among all nodes are nearly the same over different numbers of requests.

relayers also increases when the number of nodes grows in the network. Since we set the same density of resellers in these simulation sets, the cost and revenue are mostly the same for a fixed set of resellers. The resellers cannot get much more profit when the size of the network is larger. However, resellers reduce the total communication cost in the network by about 20 percent. For instance, when the network size is 175, the total communication cost drops from 1889 to 1494. Meanwhile, since any node can become a relayer, it will incentivize more nodes to relay data, thus giving more revenue to relayers. One interesting discovery is that when the number of nodes in the network is smaller, (e.g., less than 50 nodes in the network in the current parameter setting), it is hard to get enough incentives for nodes to relay data. Thus, only a few relayers participate in the process. For the data producer, it will serve the consumer without resellers or relayers. Thus, the producer will gain more revenue from a larger number of requests in these situations. When the network is larger, more nodes will serve as relayers and make a profit, and share more profit with the producer. Overall, the producer receives 6.5 percent more profit with the help of resellers and 16.2 percent more profit with the help of both resellers and relayers.

Fig. 6c shows the revenue sharing ratio for relayers under different sizes of the network. The revenue sharing ratio that relayers require increases as the network size grows larger. When there are in total 75 nodes in the network, 60 percent of participating nodes require a revenue sharing ratio of 0.4 or less, but it requires a revenue sharing ratio up to 0.6 for the same fraction of nodes when the number of nodes reaches 150. Fig. 6d shows the number of resellers and relayers in the network. As the density of the nodes in each network is the same, the area is larger when there are more nodes, and it requires more cost for relayers to deliver data items. Relayers also require a much higher revenue sharing ratio when the network is larger to compensate for higher costs. Thus, profits most go to relayers, and even the producer does not get much more from the relaying processes.

4.3 Profit on Different Numbers of Requests

Next, we evaluate the performance under different numbers of requests for a data item. We set the number of requests from 20 to 80 respectively with 100 nodes in total. There are 15 resellers and the price of the data item is 50.

Fig. 7 indicates the profit for the producer (a) and other nodes (b) under different numbers of requests. More requests mean more sales, which will incentivize more nodes to sell data for profit. The profit of the producer increases as more requests in the network almost linearly, as well as the profit with the help of resellers/relayers. The costs are mostly constant but the number of data items sold increases. Meanwhile, with a larger number of requests, the profit of resellers and relayers increases. However, the increase of profit from relayers is not significant. After the number of requests larger than 50, the profit of the relayers is similar, and has some fluctuations due to randomness. Overall, the producer receives 11.4 percent more profit with the help of resellers and 25.7 percent more profit with the help of both resellers and relayers.

Fig. 7c shows the distribution of revenue sharing ratio under different numbers of requests in the network, and Fig. 7d shows the number of resellers and relayers in the network. There is no significant difference in the revenue sharing ratio and the number of participate relayers among different numbers of requests. This shows that more requests do not incentivize more nodes to become relayers. For the same number of nodes and densities, the cost for data delivery is mostly the same for a data item, and relayers need similar shares of revenue to compensate for the cost. The increasing profit comes from selling more data items for the producer. Thus, the growth rate of the profit of

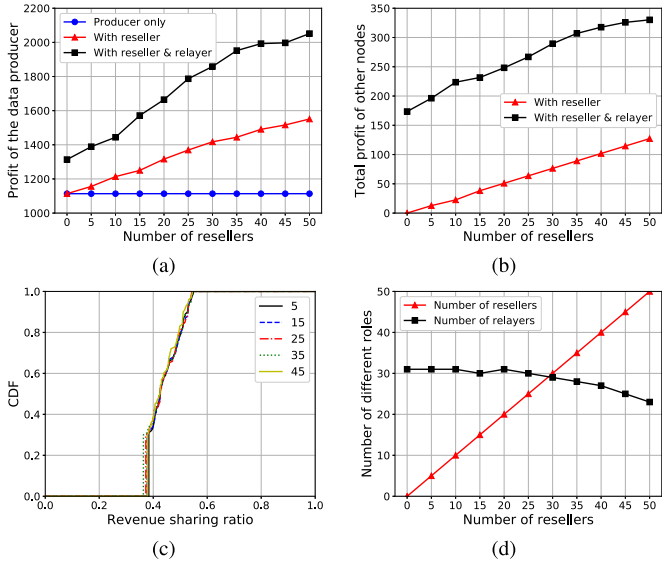


Fig. 8. The profit of producer (a) and resellers (b) under different numbers of resellers. The profit for the producer increases when there are more resellers in the network. The distribution of revenue sharing ratio under different sizes of the network is shown in (c). The number of resellers and relayers is shown in (d). The revenue sharing ratio among all nodes is nearly the same over different numbers of resellers, and the number of relayers is almost the same.

the producer is relatively larger than that of resellers and relayers, and relayers do not get much profit increase when the number of requests is larger.

4.4 Profit Over Different Numbers of Resellers

Finally, we evaluate the profit of each party under different numbers of resellers in the network. These resellers are previous consumers which already have the data item and can deliver the data item to nearby new consumers which require the data item. We test for 0 to 50 resellers in the network of 100 nodes, and the price of the data item is 50.

Fig. 8 indicates the profit for the producer (a) and other nodes (b) for different numbers of resellers. "Producer only" in Fig. 8a does not use resellers to deliver data, so it will not change under different numbers of resellers. The producer can get more profit when resellers and relayers participate and deliver data items. And the more resellers are in the network, the higher the profit the producer will get. Meanwhile, when resellers, which have obtained data in previous rounds, participate in the process, some of them may be close to some demand than the producer. Thus, using these resellers to deliver data will reduce the cost of delivery. The total profit for resellers also increases. Since the total number of demands is the same, increasing the number of resellers in the same area will result in more resellers delivering data items to nearby demands. Meanwhile, when relayers participate, having more resellers does not bring much more profit for the relayers. Since there are a certain number of requests, i.e., certain total revenue, more resellers actually take a share, and the profit of relayers will not grow super linearly as the number of resellers grows. Thus, the difference between the profit of only resellers and both relayers and resellers participating is nearly constant. Overall, the producer receives 16.6 percent more profit with the help of resellers and 49.5 percent more profit with the help of both resellers and relayers.

Fig. 8c shows the distribution of revenue sharing ratio under different numbers of resellers, and Fig. 8d shows the number of resellers and relayers in the network. The number of relayers decreases as the number of resellers goes up. This is because the number of requests is the same, as well as the total revenue. Thus, more resellers can deliver more data, and not that many relayers are needed. This explains that the difference in profit of the producer decreases as the number of resellers increases, especially when the number is large (e.g., more than 25 resellers shown in Fig. 8d).

5 Related Work

On the contrary to cloud computing which moves the computing to the centralized cloud, edge computing moves the computing work to distributed nodes on the edge of the network. The computing mostly or entirely happens on nodes near to or inside the edge devices [13]. Edge computing can offer fast and robust data sharing and processing capabilities for end devices. One major research aspect of edge computing studies the benefit of using smaller edge servers (cloudlets) deploying near the network edge (e.g., cellular base stations), serving as the middle layer between edge devices and clouds [14], [15]. These edge servers can offer multiple applications such as caching and resource virtualization. Another research aspect on edge computing studies the innovative functionalities from the collaboration of edge devices. In such scenarios, the resources of nodes are often limited. This collaboration is important especially over high mobility and frequent topology change scenarios like vehicle networks [16], [17], in which the connection to even a small station is not stable.

To tackle the complicated collaboration in the edge networks, game theory is a promising technique that has been widely adopted in various networks. In [18], the authors consider a D2D communication framework in which the operator of the base station offers incentives to owners of devices to motivate content communication. In [19], a wireless sensor network consisting of many private sensor networks is considered. Shen *et al.* [20] use the Stackelberg game to incentivize content providers in the wireless networks to participate in caching, which can improve the QoS for wireless devices of small base stations. Our work uses similar strategies to incentivize nodes to serve as relayers while focusing on revenue sharing among intermediate parties.

Blockchain technology is proposed in 2008 by Satoshi Nakamoto [21] and has been widely used in cryptocurrencies. A blockchain consists of multiple blocks, each of which contains the information of previous blocks to form a chain. The blockchain is designed to prevent unauthorized changes of its contents using cryptography features. If a malicious user wants to tamper with a piece of data, it has to counterfeit a whole branch of chain from the block that it intends to modify. The time and energy to counterfeit a chain are far over the benefit a malicious party can get. A typical cryptography feature is called Proof of Work (PoW) [22], [23], which requires nodes to find a certain hash pattern of certain data. This makes it easy to verify but hard to counterfeit a data item in the blockchain. These features can make the

blockchain system a safe ledger perfectly for cryptocurrencies, e.g., Bitcoin [21], Litecoin [24], and Ethereum [25], [26].

The concept of the smart contract is proposed by Nick Szabo [7] and becomes a reality implemented in the core of Ethereum [25]. It is a computer protocol that makes sure that the negotiation and implementation of a contract can be enforced without a trusted third party as the arbitrament. Blockchain technology makes the smart contract practical. Contacts are publicly stored in the blocks that all nodes in the blockchain can access, and the corresponding transactions are irreversible. Thus, many applications are emerged using the smart contract concept, e.g., anonymous voting [27] and private IoTs [28].

Copyright protection is a key factor for data trading. The benefit of the data producer will be violated if the data can be copied and redistributed without the permission of data the data owner. Digital rights management (DRM) is proposed to prevent digital data from unauthorized redistribution, like Microsoft DRM [29] and Apple HLS DRM [30]. This is good to protect structured data such as software or multimedia, but it is hard to detect deliberate replication and tampering of unstructured data, such as docs and PDF. Thus, in most cases, data is simply not allowed to be redistributed or resold for data trading [31], [32].

Data trading with the blockchain technology in edge scenarios also gains much attention from researchers these years. Chen *et al*. [33] propose a blockchain-based data trading approach for vehicle networks. They use a hierarchal structure to ensure secure data trading at the blockchain level and design an iterative double auction to get proper data prices. Missier *et al*. [34] propose a blockchain and use smart contract to create a decentralized trading platform for IoT data, and implement proof-of-concept trading systems. Our work focuses more on sharing the revenue with intermediate parties which can increase the security and availability of traded data and uses smart contracts to ensure the revenue is shared properly among parties. Meanwhile, Jung *et al*. [35] propose a series of secure protocols against dishonest consumers from reselling data, but a centralized broker may be needed. Our work focuses on a fully distributed mechanism where there is no centralized control or trusted third parties needed.

6 Conclusion and Discussion

In this paper, we have proposed a profit protecting trading mechanism for the data producer and resellers/relayers in pervasive edge computing environments. We have proposed a smart-contract based protocol to ensure the profit of the producer for reselling, and we have proposed a protocol to detect unauthorized reselling without imposing extra burdens onto the consumer and the producer. We have formulated a two-stage Stackelberg game to find a revenue sharing ratio between the data producer and resellers/relayers, and have proved the approximation ratio between the rounded result and the optimal integer result. Simulation results have shown that our proposed mechanism works better than that without reselling and relaying processes under pervasive edge computing environments.

Our solution is not a typical way to solve the Stackelberg game. The producer takes final decisions on which node stores which data item based on the response of different resellers/relayers. If there are more than one reseller or relayer offer a profitable revenue sharing ratio, the producer will reduce the revenue shared until only one reseller or relayer is left profitable to deliver the data. This can prevent duplicate transmissions and give the producer more profit, yet the producer needs to proactively solve the problem.

Peer data trading is an important application in edge environments. IoT sensing data, "We media" contents, and other valuable information can be traded among edge devices and servers. Our proposed solution enables trading parties to get protected profit shares. It supports safe transactions in distributed and untrusted environments. The edge scenario is a representative instance in which such data trading constantly happens.

Smart-contract is a well-known protocol to enforce the execution of a certain contract. We use smart-contract to make sure the revenue is properly shared and each party pays and gets its corresponding share. Currently, some cryptocurrencies like Ethereum [25] have implemented smart-contract and some previous work like [18] have evaluated its performance. Evaluating smart-contract security and performance is not in the scope of this paper.

There are some limitations of the current model that demand future research. In the reselling process, the reseller can add noise to the original content to make it harder to be detected. Consumers can choose a relatively small part of the data item for validation, which can reduce the influence of the noises. However, this can increase the possibility of collisions to other data items, which may trigger a false positive response for unauthorized reselling. A simple solution is for consumers to conduct the verification process under all conditions (including buying from legal resellers), which can be used to trace the root for unauthorized reselling. We will discuss the relationship between the noise and the successful verification ratio and how to reduce the false-positive ratio under such conditions in our future work.

Acknowledgments

This work was supported in part by the US National Science Foundation under Grants 1513719 and 1730291.

References

[1] "Gumroad," Accessed: Jul. 17, 2019. [Online]. Available: https://gumroad.com/

[2] "Is gumroad a scam?," Accessed: Jul. 17, 2019. [Online]. Available: https://www.quora.com/Is-Gumroad-a-scam

[3] H. Green, "Theft, lies, and facebook video," 2015. Accessed: Jul. 17, 2019. [Online]. Available: https://medium.com/@hankgreen/theft-lies-and-facebook-video-656b0ffed369

[4] Y. Huang, J. Zhang, J. Duan, B. Xiao, F. Ye, and Y. Yang, "Resource allocation and consensus on edge blockchain in pervasive edge computing environments," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst.*, 2019, pp. 1476–1486.

[5] Z. Huang, X. Su, Y. Zhang, C. Shi, H. Zhang, and L. Xie, "A decentralized solution for IoT data trusted exchange based-on blockchain," in *Proc. 3rd IEEE Int. Conf. Comput. Commun.*, 2017, pp. 1180–1184.

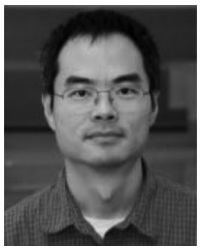
[6] Y. Huang, X. Song, F. Ye, Y. Yang, and X. Li, "Fair and efficient caching algorithms and strategies for peer data sharing in pervasive edge computing environments," *IEEE Trans. Mobile Comput.*, vol. 19, no. 4, pp. 852–864, Apr. 2020.

[7] N. Szabo, "Formalizing and securing relationships on public networks," *First Monday*, vol. 2, no. 9, Sep. 1997, doi: 10.5210/fm.v2i9.548.

[8] K. Poularakis, G. Iosifidis, I. Pefkianakis, L. Tassiulas, and M. May, "Mobile data offloading through caching in residential 802.11 wireless networks," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 1, pp. 71–84, Mar. 2016.
[9] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[10] R. Hemmecke, M. Köppe, J. Lee, and R. Weismantel, "Nonlinear integer programming," in *50 Years of Integer Programming 1958–2008*. Berlin, Germany: Springer, 2010, pp. 561–618.
[11] J. Lee and S. Leyffer, *Mixed Integer Nonlinear Programming*. Berlin, Germany: Springer, 2011.
[12] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *J. Mach. Learn. Res.*, vol. 17, no. 83, pp. 2909–2913, 2016.
[13] Microsoft Research, "Edge computing," 2008. Accessed: Oct. 10, 2020. [Online]. Available: https://www.microsoft.com/en-us/research/project/edge-computing/
[14] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-ased cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Fourth Quarter 2009.
[15] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.
[16] D. Liu, B. Chen, C. Yang, and A. F. Molisch, "Caching at the wireless edge: Design aspects, challenges, and future directions," *IEEE Commun. Mag.*, vol. 54, no. 9, pp. 22–28, Sep. 2016.
[17] P. Garcia Lopez *et al.*, "Edge-centric computing: Vision and challenges," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 5, pp. 37–42, 2015.
[18] Z. Chen, Y. Liu, B. Zhou, and M. Tao, "Caching incentive design in wireless D2D networks: A stackelberg game approach," in *Proc. IEEE Int. Conf. Commun.*, 2016, pp. 1–6.
[19] Y. Zeng, P. Zhou, J. Liu, and Y. Yang, "A stackelberg game framework for mobile data gathering in leasing residential sensor networks," in *Proc. IEEE/ACM 26th Int. Symp. Qual. Service*, 2018, pp. 1–6.
[20] F. Shen, K. Hamidouche, E. Bastug, and M. Debbah, "A stackelberg game for incentive proactive caching mechanisms in wireless networks," in *Proc. IEEE Global Commun. Conf.*, 2016, pp. 1–6.
[21] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2019. [Online]. Available: https://bitcoin.org/bitcoin.pdf
[22] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 3–16.
[23] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "SoK Research perspectives and challenges for bitcoin and cryptocurrencies," in *Proc. IEEE Symp. Secur. Privacy*, 2015, pp. 104–121.
[24] K. Fanning and D. P. Centers, "Blockchain and its coming impact on financial services," *J. Corporate Accounting Finance*, vol. 27, no. 5, pp. 53–57, 2016.
[25] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *White Paper*, vol. 3, no. 37, 2014.
[26] T. Chen *et al.*, "Understanding ethereum via graph analysis," in *Proc. IEEE Conf. Comput. Commun.*, 2018, pp. 1484–1492.
[27] P. McCorry, S. F. Shahandashti, and F. Hao, "A smart contract for boardroom voting with maximum voter privacy," in *Proc. Int. Conf. Financial Cryptography Data Security*, 2017, pp. 357–375.
[28] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.
[29] D. K. Mulligan, J. Han, and A. J. Burstein, "How DRM-based content delivery systems disrupt expectations of personal use," in *Proc. 3rd ACM Workshop Digital Rights Manage.*, 2003, pp. 77–89.
[30] C. D'Orazio and K.-K. R. Choo, "An adversary model to evaluate DRM protection of video contents on IoS devices," *Comput. Secur.*, vol. 56, pp. 94–110, 2016.
[31] F. Liang, W. Yu, D. An, Q. Yang, X. Fu, and W. Zhao, "A survey on big data market: Pricing, trading and protection," *IEEE Access*, vol. 6, pp. 15 132–15 154, 2018.
[32] X.-Y. Li, J. Qian, and X. Wang, "Can china lead the development of data trading and sharing markets?," *Commun. ACM*, vol. 61, no. 11, pp. 50–51, 2018.
[33] C. Chen, J. Wu, H. Lin, W. Chen, and Z. Zheng, "A secure and efficient blockchain-based data trading approach for Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 9110–9121, Sep. 2019.
[34] P. Missier, S. Bajoudah, A. Capossele, A. Gaglione, and M. Nati, "Mind my value: A decentralized infrastructure for fair and trusted IoT data trading," in *Proc. 7th Int. Conf. Internet Things*, 2017, pp. 1–8.
[35] T. Jung *et al.*, "Accounttrade: Accountable protocols for big data trading against dishonest consumers," in *Proc. IEEE Conf. Comput. Commun.*, 2017, pp. 1–9.

Yaodong Huang received the BE degree in computer science and technology from the University of Electronic Science and Technology of China, in 2015. He is currently working toward the PhD degree in computer engineering at Stony Brook University. His research interests include mobile edge computing, including caching, security, privacy, and energy-efficiency in mobile edge environments.

Yiming Zeng received the BEng degree in information engineering from the Shanghai Jiao Tong University, Shanghai, China. He is currently working toward the PhD degree in computer and electrical engineering at Stony Brook University, New York, NY. His research interests include addressing computing, privacy, and caching issues in edge networks.



Fan Ye received the PhD degree from Computer Science Department, UCLA, in 2004. He is currently an associate professor at the ECE Department of Stony Brook University, before that he was a research staff member with IBM T. J. Watson Research. His research interests include mobile sensing platforms, systems and applications in healthcare and location based services, edge computing, Internet-of-Things, and data-centric wireless communication. He has published more than 100 papers with more than 12,000 citations according to Google Scholar, and 30 granted/pending patents/applications. He has received NSF CAREER award, Google Faculty Research Award, IBM Research Division Award and 5 Invention Achievement Plateau awards, Best Paper Award for IEEE ICCP 2008. He has been a panelist for NSF, NIH and Canada, Hong Kong government funding agencies, on program/organizing committees for conferences including IEEE Infocom, IEEE ICDCS, ACM Mobicom, ACM Sensys.

Yuanyuan Yang (Fellow, IEEE) received the BEng and MS degrees in computer science and engineering from Tsinghua University, Beijing, China, and the MSE and PhD degrees in computer science from Johns Hopkins University, Baltimore, Maryland. She is currently a SUNY distinguished professor of computer engineering and computer science at Stony Brook University, New York, and is currently on leave at the National Science Foundation as a program director. Her research interests include edge computing, data center networks, cloud computing, and wireless networks. She has published about 400 papers in major journals and refereed conference proceedings and holds seven US patents in these areas. She is currently the associate editor-in-chief for the *IEEE Transactions on Cloud Computing* and an associate editor for ACM Computing Surveys. She has served as an associate editor-in-chief and associate editor for the *IEEE Transactions on Computers* and associate editor for the *IEEE Transactions on Parallel and Distributed Systems*. She has also served as a general chair, program chair, or vice chair for several major conferences and a program committee member for numerous conferences.