Fair and Protected Profit Sharing for Data Trading in Pervasive Edge Computing Environments

Yaodong Huang, Yiming Zeng, Fan Ye, Yuanyuan Yang
Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY 11794, USA
{yaodong.huang, yiming.zeng, fan.ye, yuanyuan.yang}@stonybook.edu

***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 formulate a fair revenue sharing problem to maximize the profit of both the data producer and resellers. 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. Extensive simulations show that with resellers, our mechanism can achieve higher profit for the data producer and resellers.**

***Index Terms*—Pervasive edge computing, Blockchain, Smart contract, Game theory**

I. 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 the abundance of devices and data. Producers, owner of certain devices, can provide information or services to consumers for incomes. One example is "We Media", where data producers trade their content, mainly video clips or texts, to other consumers and make money.

Consider a situation where the producer has for-profit contents 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 contents and subscriptions. For example, Gumroad [1] provides this kind of model for producers to sell 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 the blockchain technology for data trading in edge environments. The blockchain technology used widely in cryptocurrencies is a secure ledger for sharing micro-payment and micro-access control information in such distributed environments. There are many security features built in the blockchain. First, the completed transaction history is encoded in the blockchain for quick 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. The 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 in such distributed systems, there are many challenges in 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 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.

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 revenue sharing between different nodes is 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 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 higher profit for the producer and also share it among 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 consumer and producer.
- We model the interaction between the producer and resellers as a two-stage dynamic Stackelberg game to determine the optimal revenue sharing ratio and derive the analytical solutions for both the producer and resellers. We analyze the data storage and delivering scheme and prove that an approximation ratio exists between the rounded equilibrium of the Stackelberg game and the optimal result.
- We implement our proposed mechanism and conduct extensive evaluation. The results show that the proposed mechanism can achieve more profit for the producer with the help of resellers, while offering profit for resellers to incentivize participation in the network.

II. Reselling Process and Game Formulation

In this section, we introduce the data trading model and the Stackelberg game formulation. We discuss the reselling process as well as the revenue sharing mechanism.

TABLE I
NOTATIONS USED IN THE PAPER

$i \in \mathcal{I}$	Reseller nodes and reseller 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
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

There are three 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. We only consider one producer in our problem who generates the original data items for selling.

A. Reselling Process

Nodes can take different roles 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 the other, 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. Once it receives data item k, the 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 the proof of data ownership and rejects the transaction. The detailed design is presented in Section III.

B. Revenue Sharing Game Formulation

As we mentioned above, the producer shares the revenue with resellers at a sharing ratio r_{ik} when they are involved in this selling process. We define the revenue sharing ratio between the producer and resellers as the ratio of the revenue (i.e., price) the resellers gets 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 both parties. By offering a fair ratio, the producer can encourage more resellers to participate and achieve the maximal profit for both producer and resellers. For each producer, the profit comes from the shared revenue of resellers and the sale directly to consumers. The formulation is as follows.

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

$$\text{s.t.} \quad \sum_{i\in\mathcal{I}} y_{ijk} + y_{0jk} = \rho_{jk}, \ (\forall j \in \mathcal{J}, \forall k \in \mathcal{K}) \tag{2}$$

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

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

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 node i will send 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 directly for selling data item k to node j, and y_{0jk} determines whether the consumer needs to send the data item to node j. 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. 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.

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

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

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

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} is a determination variable, where $x_{ik} = 1$ indicates the 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 [4], we denote the cost of for 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 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.

III. Reselling Protocol Design

In this section, we describe our reselling protocols. We present the processes under authorized and unauthorized reselling situations and detection of unauthorized reselling.

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 fair sharing of their profit. There are small amount of rogue nodes which 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 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. 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 unauthorized process can be detected and dealt with.

A. 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 [5] is a protocol that once it is signed by involved parties, the contract content can be viewed and validated by others. Smart contract is then encoded in blocks and transactions 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 public keys. The contract now takes effect and is stored into blockchain to be publicly validated. Fig. 1 shows the reselling process.

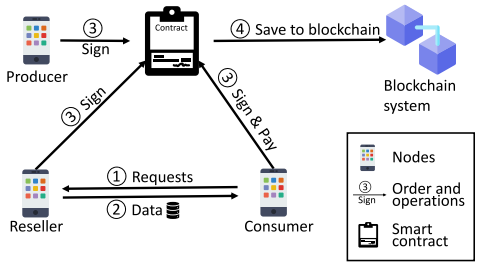


Fig. 1. Step by step information for an authorized reselling process. 1) The consumer requests a data item from the nearby reseller. 2) The reseller then sends the data to the consumer. 3) The reseller generates a smart contract. The producer, consumer, and reseller all need to sign the contract to make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

Since the contract takes effect only when the consumer, producer and reseller sign, it can avoid the 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 contact 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 can be verified. Other nodes then acknowledge and enforce the credit change of each node when the contact takes effect and stored in the blockchain. 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 is not revealed.

B. 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 that this is a new data item.

Once a consumer finds this 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 such resold without authorization, the producer will present the information to prove it. The information detail is presented in the validation process. 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 an unauthorized reselling.

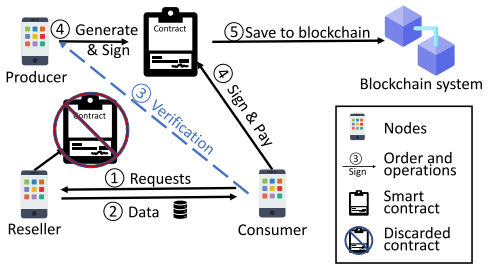


Fig. 2. Step by step information for an unauthorized reselling process. 1) The consumer requests a data item from the nearby reseller. 2) The reseller then sends the data to the consumer. 3) The consumer broadcast a piece of verification information and the data producer will notice. 4) The data producer 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 by all nodes.

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

C. Verification Process

As we mentioned above, the consumer checks whether the data item is genuine. After the consumer gets the data item which the producer is not obvious, the consumer conducts 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. 3(a). It continues to use different nonce until getting a hash value that follows a certain pattern (e.g., hash smaller than a certain number). 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. Then, the consumer broadcasts the verification packet, shown in Fig. 3(b), 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 check 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 contact 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 contact and the consumer will sign this new contract. The contract of the rogue reseller will be discarded, and the reseller will not receive any revenue.

Note that rouge consumers may deny paying 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.

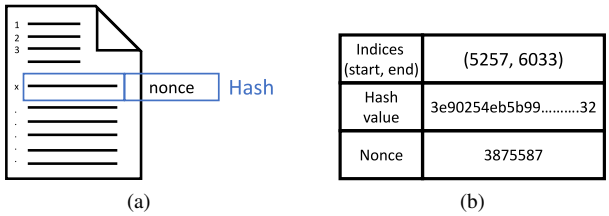


Fig. 3. The hashed item (a) and the verification packet example (b). The hash included a part form the data and a nonce to meet previous set pattern of the hash value.

During the verification process, the consumer contributes "work" to check the data item information by obtaining 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 credit, which can be used to pay for the data item. Since the credit is presented, the consumer benefit will be lost if we just cancel the transaction. Thus, we let the producer get the corresponding revenue directly from the consumer. In this case, we can keep the profit of the producer, and the "work" of the consumer is not wasted.

IV. Solutions to Revenue Sharing Game

In this section, we propose a two-stage dynami Stackelberg game to model the interaction between the producer and resellers, and provide analytical solutions to the model.

A. Stackelberg Game Model

In the game we propose, the producer is modeled as the leader, and resellers 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. In Stage II, after given incentive revenue from the producer, resellers need to determine which data items to buy and deliver to maximize their own profit. The game is defined as follows.

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

The game is a two-stage dynamic Stackelberg game with complete information, 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 reaches the equilibrium if following conditions are satisfied at the same time:

$$S(r^*, y^*) \geq S(r, y^*), (\forall r) \tag{8}$$

$$U(x^*, y^*, r^*) \geq U(x, y, r^*), (\forall x, y) \tag{9}$$

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

B. 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 are coupled together. The processes of the interaction between the producer and resellers are dynamic, the profit of the producer and resellers cannot be determined at the same time.

To analyze the problem, we separate the process of the game in two different stages. In Stage I, the producer decides the revenue sharing ratio offered to resellers to encourage them to buy and deliver data items. In return, for resellers, they 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 determination.

1) Stage II: We first address the case in Stage II. The objective for resellers is to maximize their total profit. After observing the action of the leader (the producer offers the incentive ratio), determination of resellers is decided 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 is defined as

$$\max_{x,y} U_i(x, y), \tag{10}$$
$$\text{s.t.}(6), (7).$$

The objective function of the problem is discrete due to the discrete variable of x_{ik}. To solve the reseller problem, the discrete storage variable x_{ik} is relaxed from {0,1} to [0,1] which is continuous. Hence, the profit function (5) is monotone increasing with the determination variable y_{ijk}. To maximize the profit of the reseller, the optimal solution achieves when (6) transfers as follows,

$$x_{ik}\rho_{jk} = y_{ijk}.(\forall i \in \mathcal{I} \cup \mathcal{I}', \forall j \in \mathcal{J}, \forall k \in \mathcal{K}). \tag{11}$$

The determination variable y_{ijk} is replaced from (11). After replacing the determination variable y_{ijk}, the profit function of the reseller (5) is continuous about the x_{ik}. We first calculate the partial maximization over x of the profit function. The first derivative function is derived as follows,

$$\frac{\partial U(x, y)}{\partial x_{ik}} = \sum_j r_{ik}p_k\rho_{jk} - \sum_j c_{ijk}\rho_{jk} - p_k - \frac{o_k l_{ik}}{1 + o_k - x_{ik}o_k}. \tag{12}$$

Equation (12) is a concave function. To get the maximization value, let $\frac{\partial U(x,y)}{\partial x_{ik}} = 0$, we have

$$x^*_{ik} = -\frac{l_{ik}}{r_{ik}p_k \sum_j \rho_{jk} - \sum_j c_{ijk}\rho_{jk} - p_k} + 1 + \frac{1}{o_k}. \tag{13}$$

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

$$r_{ik}p_k \sum_j \rho_{jk} - \sum_j c_{ijk}\rho_{jk} - p_k \leq l_{ik}o_{ik}, \tag{14}$$

$$(r_{ik}p_k \sum_j \rho_{jk} - \sum_j c_{ijk}\rho_{jk} - p_k)(o_k + 1) \geq l_{ik}o_{ik}, \tag{15}$$

where (14) corresponds to the condition $x^*_{ik} \leq 1$ and (15) indicates that $x^*_{ik} \geq 0$. If these conditions cannot be satisfied, the optimal solution is reached either $x^*_{ik} = 0$ or $x^*_{ik} = 1$. To solve this problem, the storage variable is relaxed from $\{0, 1\}$ to $[0, 1]$, the solution derived is not feasible to the original problem. Hence, we propose a rounding policy to round the continuous $[0, 1]$ back to discrete $\{0, 1\}$. We define λ as the boundary value. If $x_{ik} \geq \lambda$, $x_{ik} = 1$, otherwise, $x_{ik} = 0$. The performance guarantee is discussed in Section IV-C.

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.

2) Stage I: Now we discuss 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 (13) and constraints (14), (15), the problem to maximize the revenue of the producer can be formulated as follows,

$$\max_r S(y, r), \tag{16}$$

$$s.t. (2), (11), (13), (14), (15).$$

By implementing constraints (2), (11) and (13), we eliminate the determination variable y_{0jk}, y_{ijk} and x_{ik}. After that, the revenue function only contains the variable r_{ik} as shown in the following,

$$S(r) = \sum_i \sum_k (l_{ik}(1 + \frac{\sum_j c_{ijk}\rho_{jk} + p_k - \sum_j c_{0jk}\rho_{jk}}{r_{ik}p_k \sum_j \rho_{jk} - \sum_j c_{ijk}\rho_{jk} - p_k})) \quad (17)$$

$$+ \sum_i \sum_k \frac{(o_k + 1)(1 - r_{ik})p_k \sum_j \rho_{jk}}{o_k} \quad (18)$$

$$+ \sum_j \sum_k (p_k - c_{0jk})\rho_{jk} \quad (19)$$

$$- \sum_i \sum_j \sum_k \frac{(o_k + 1)(p_k - c_{0jk})\rho_{jk}}{o_k}. \quad (20)$$

The objective function is not a standard convex optimization problem, the convexity or concavity of the function depends on the parameter settings which corresponds to a different solution. (18) is a linear function about r_{ik}, (19) and (20) only contain the constant parameters, they do not affect the convexity of the revenue function. The convexity of (17) varies with the parameter settings. We discuss different conditions for each i and k in detail as follows.

1) $\sum_j c_{0jk}\rho_{jk} - \sum_j c_{ijk}\rho_{jk} - p_k > 0$.
The revenue function (17) is concave about the r_{ik}, 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 [6] can be applied.
2) $\sum_j c_{0jk}\rho_{jk} - \sum_j c_{ijk}\rho_{jk} - p_k < 0$.
The revenue function (17) is a convex function about r_{ik}. The maximum value is determined by the end point of the r_{ik} which is 0 or 1. In this case, the optimal strategy of the reseller can be derived by comparing the revenue value of the objective function between $r_{ik} = 0$ and $r_{ik} = 1$. When $r_{ik} = 1$, which means the producer does not need to serve consumers directly, resellers can satisfy all requests from consumers and it will cost more for the producer to serve consumers directly. When $r_{ik} = 0$, the revenue offered to resellers is larger than the producer serving the requests from consumers directly. Thus, the best strategy of the producer is to serve consumers directly.
3) $\sum_j c_{0jk}\rho_{jk} - \sum_j c_{ik}\rho_{jk} - p_k = 0$.
The revenue function (17) is monotone decreasing about the revenue sharing ratio r_{ik}. Hence, the optimal solution is $r^*_{ik} = 0$. As mentioned above, the producer can get the largest income when $r_{ik} = 0$, which means the optimal strategy for the producer is to sell the data item by itself instead of offering the incentives to resellers.

C. Performance Analysis

In our proposed game, storage variables x_{ik} and y_{ijk} are integers. Directly solving the problem to get optimal results as integers is difficult as the problem is NP-Hard [4]. Thus, in Stage II, storage variables x_{ik} and y_{ijk} 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 rounding policy and prove that it has an approximation ratio to the optimal result.

Rounding Policy. If $x_{ik} \geq \lambda$, $x_{ik} = 1$, otherwise, $x_{ik} = 0$, where λ is the rounding threshold.

To simplify the proof process, we denote the result of optimization problem (5) as $U^*(x)$ with optimal integer result, and $U(x)$ with the relaxed (i.e., relaxed $x^*_{ik} \in [0, 1]$). $U^\dagger(x)$ is the result after rounding (i.e., integer $x^\dagger_{ik} \in \{0, 1\}$). The theoretical bound for the rounding policy is derived as follows.

Theorem 1. The rounding policy is an approximation algorithm to the original problem without rounding and it achieves an approximation ratio of $\frac{1}{\lambda(1-\epsilon)}$, i.e., $U^*(x) \leqslant \frac{1}{\lambda(1-\epsilon)} U^\dagger(x)$.

Proof. The objective function (5) contains the 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.

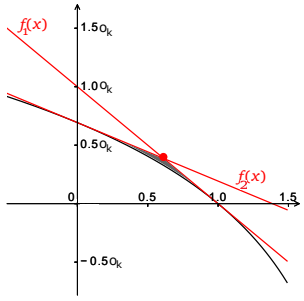


Fig. 4. The relaxation of the non-linear part of the objective function. $f_1(x)$ and $f_2(x)$ are two tangent lines which intersect the non-linear funtion at integer points.

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 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,

$$f_1(x) = -o_k x + o_k, \quad (21)$$

$$f_2(x) = -\frac{o_k}{1 + o_k} x + log(1 + o_k). \quad (22)$$

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

After relaxing of the logarithmic function into linear functions, $U_i(x)$ can be written as a general linear form $V_i(x) = d_{ik}x_{ik}$, where $U_i(x) \leqslant V_i(x)$, and d_{ik} is regarded as the summation of all parameters. Hence, the total revenue function can be written as $V(x) = \sum_i d_{ik}x_{ik}$. This theorem can be proved according the following inequality deduction,

$$
\begin{aligned}
U^*(x) &\leqslant U(x) \leqslant V(x) \\
&= \sum_i d_{ik} x_{ik} && (23)\\
&= \sum_{\{x_{ik}|x_{ik}\geqslant\lambda\}} d_{ik}x_{ik} + \sum_{\{x_{ik}|x_{ik}<\lambda\}} d_{ik}x_{ik} && (24)\\
&= \frac{1}{\lambda}\sum_{\{x_{ik}|x_{ik}\geqslant\lambda\}} d_{ik}x_{ik} + (1-\frac{1}{\lambda})\sum_{\{x_{ik}|x_{ik}\geqslant\lambda\}} d_{ik}x_{ik} \\
&+ \sum_{\{x_{ik}|x_{ik}<\lambda\}} d_{ik}x_{ik} && (25)\\
&\leqslant \frac{1}{\lambda}\sum_{\{x_{ik}|x_{ik}\geqslant\lambda\}} d_{ik} + \sum_{\{x_{ik}|x_{ik}<\lambda\}} d_{ik}x_{ik} && (26)\\
&\leqslant \frac{1}{\lambda}\sum_{\{x_{ik}|x_{ik}\geqslant\lambda\}} d_{ik} + \epsilon\sum_i d_{ik}x_{ik} && (27)\\
&= \frac{1}{\lambda}V^\dagger(x) + \epsilon V(x). && (28)
\end{aligned}
$$

We explain the deduction step by step. (23) to (25) are identical transformations to make $V^\dagger(x)$. Inequality (26) can be obtained because after the rounding. Inequality (27) holds since $\{x_{ik}|x_{ik} < \lambda\}$ is the subset of the reseller set, and $\epsilon < 1$. (28) is equivalent to (27). Since $V^\dagger(x) = U^\dagger(x)$ when using same rounding policy to $\{0, 1\}$, and $U^*(x) \leqslant V(x)$, we can get

$$U^*(x) \leqslant \frac{1}{\lambda(1-\epsilon)}U^\dagger(x). \quad (29)$$

We consider two linear relaxations to the logarithmic function, and different choices of λ will generate different gaps. The feasible gap region is the shadow area in Fig. 4. The intersection point x_l is the boundary condition of two different linear relaxations when solved together. Thus, when $\lambda = x_l$, it indicates the upper bound of the gap between relaxation function and original function. □

V. 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 sharing between the data producer and resellers on different settings of the network, and whether the reselling makes more profit for nodes. Profit functions are defined in (1) and (5) respectively for the producer and resellers.

In the simulation, we assume that nodes are distributed randomly in a square area with the same density of $25m^2$ unit per nodes. 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. "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. Note that every node can be a reseller if the revenue sharing ratio r_{ik} is reasonable and profitable. Without loss of generality, we only consider selling one data item over the network. We implement the Stackelberg game using python. The convex optimization problem (17)-(20) is solved using CVXPY [7]. We conduct our simulations on a computer with an Intel Core i7-5820K processor and 32GB RAM.

A. Profit on Different Prices of Data Items

We first evaluate the profit that the data producer and resellers can get under different prices of data items. We set 100 nodes in the area. 50 consumers are demanding the data item, and each demand needs to be satisfied by either the producer or a reseller.

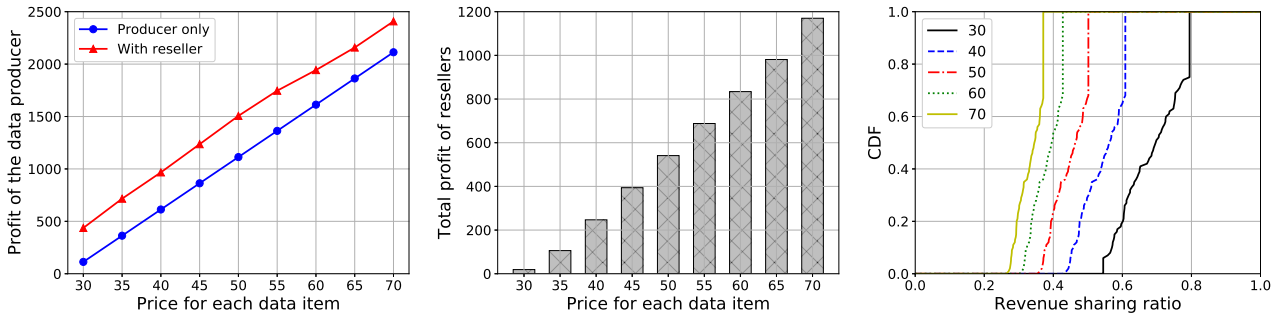
Fig. 5 shows the profit of the producer (a) and resellers (b) over different prices of data items. In general, the higher the price is, the higher the profit will be for all nodes. Since demands are fixed and the communication costs are almost constant, the profit grows almost perfectly linearly with the increasing price of the data item. Meanwhile, if resellers join, the producer will receive more profit. Since more nodes store the data item, and a consumer can access data from a nearby node, the communication cost is lower for delivering data. The reselling nodes can have more profit, which also increases the profit for the producer. Overall, the producer receives 30.9% more profit with the help of resellers.

Fig. 5(c) shows the CDF of different revenue sharing ratio r_{ik} which reseller i gets the ratio 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 resellers because they gain more than the cost for data delivery. At the same revenue sharing ratio, the higher the CDF, the more nodes will participate in the reselling process. For instance, when the price of the data item is 50, 20% of the revenue sharing ratio is less than 0.4, i.e., only 20% nodes will participate reselling if it can get 40% of the overall price. When the price increases to 60, 50% nodes will participate at the same ratio of the overall price. Since all nodes are selfish and rational, they will try to compensate their cost for data delivery if they participate in the reselling. Thus, if the price is high, reselling nodes can easily compensate the cost, and resellers can require a lower revenue sharing ratio and still make profit, and more nodes will resell data. This in turn gives more profit of the producers. The higher profit for producer shows that our proposed mechanism indeed helps to increase the benefit of the producer.

B. Profit on Different Sizes of Networks

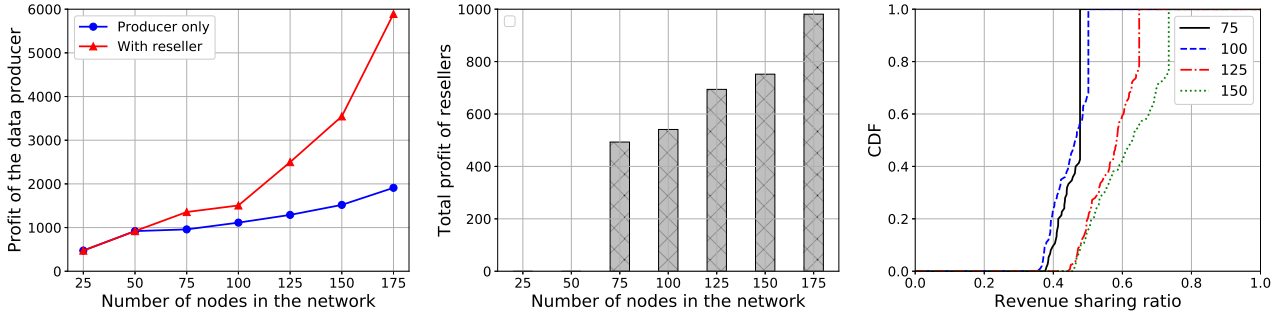
Next, we evaluate the profit of the data producer and resellers under different sizes of networks. We set 25 to 175 nodes in the same area. The price of the data item is 50, and there are randomly 50% nodes requesting the data item.

Fig. 6 shows the profit for the data producer (a) and resellers (b). In general, the profit of producers increases as more nodes are in the network. Since more nodes bring more requests, the producer can make more profit by selling more data items. An interesting discovery is that when there are fewer than



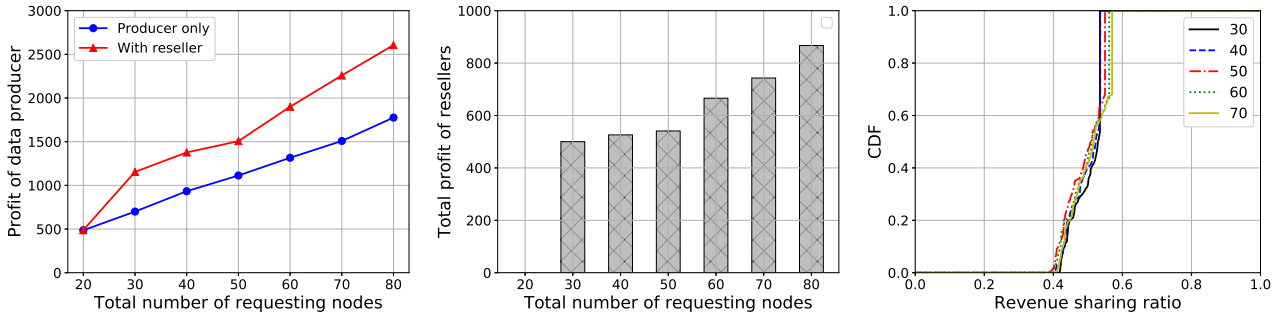
(a) (b) (c)

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 higher the price is, the more nodes will participate and require less revenue sharing ratio.



(a) (b) (c)

Fig. 6. The profit of the producer (a) and resellers (b) under different sizes of the network. The profit for producer grows as networks with increasing nodes. If less 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 larger the network size is, the higher sharing ratio resellers will require .



(a) (b) (c)

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

50 nodes thus fewer than 25 demands in the network, the producer does not need to share revenue with resellers. This is because resellers receive no profit under such conditions. Thus, it does not gain more profit with resellers. With such a small-sized network, the producer does not make much profit, and sharing revenues with resellers will only diminish the profit. Meanwhile, if the network size is large, the profit for the producer grows significantly larger with the help of resellers. Overall, the producer receives 97.8% more profit with the help of resellers.

Fig. 6(c) shows the revenue sharing ratio under different sizes of the network. As the network size grows larger, the revenue sharing ratio required from resellers is larger. When the total number of nodes is 70, 0.45 revenue sharing ratio can incentivize 60% nodes participate, and it requires about 0.7 revenue sharing ratio to incentivize the same fraction of nodes to participate if the number of nodes is 150. Since the density of nodes is the same, the larger number of nodes indicates the larger area. Thus, it requires more cost for resellers to deliver data items, and to get more revenue from the producer which can compensate for the higher costs.

C. Profit on Different Numbers of Requests

Finally, we evaluate the profit distribution between the producer and resellers under different numbers of requests for

a data item. We set the number of requests from 20 to 80 respectively in the same area with 100 nodes in total. The price of each data item is 50.

Fig. 7 indicates the profit for the producer (a) and resellers (b) in different numbers of requests. As there are more requests in the network, nodes can sell more data and make more profit. As we observe in Fig. 7(a), with very few demands, the revenue is not shared with resellers. The profit using only the producer to deliver is almost linear, since the costs are mostly constant. Meanwhile, with a larger number of requests, resellers will help the producer get more profit. The producer receives 44.1% more profit with the help of resellers.

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

VI. Related Work

On the contrary of 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 [8]. 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 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 [9], [10]. These edge servers can offer multiple applications such as caching and resource virtualization. Another research aspect studies the innovative functionalities from the collaboration of edge devices. In such scenarios, resources of nodes are often limited. This collaboration is important especially over high mobility and frequent topology change scenarios like vehicle networks [11], [12], in which the connection to even small station is not stable [13].

The blockchain technology is proposed in 2008 by Satoshi Nakamoto [14]. 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. These features can make the blockchain system a safe ledger perfectly for cryptocurrencies, e.g., Bitcoin [14], Litecoin [15], and Ethereum [16].

The concept of smart contract is proposed by Nick Szabo [5] and has been implemented in Ethereum [16]. It is a computer protocol that makes sure that a contract can be enforced without a trusted third party as the arbitrament. The 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 [17] and private IoTs [18].

Copyright protection is a key factor for data trading. Digital rights management (DRM) is proposed to prevent digital data from unauthorized redistribution, like Microsoft DRM [19] and Apple HLS DRM [20]. It 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 texts and codes. Thus, in most cases, data is simply not allowed to be redistributed [21], [22]. Meanwhile, Jung et al. [23] propose some protocols against dishonest consumers for 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 party needed.

VII. Conclusion and Discussion

In this paper, we have proposed a fair and data producer profit protecting data trading mechanism in pervasive edge computing environments. We have proposed a smart-contract based protocol to ensure the profit of 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 dynamic Stackelberg game to find a fair revenue sharing ratio between the data producer and the resellers, and have proved the approximation ratio between the rounded result and the optimal integer result. Simulations have show that our proposed mechanism works better than that without reselling processes under pervasive edge computing environments.

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 fair and 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 fair share. Currently, some cryptocurrencies like Ethereum [16] have implemented smart-contract and some previous work like [24] have evaluated its performance. Evaluating smart-contract security and performance is out of scope of this paper. In the reselling process, consumers can choose a smaller part to hash, which can reduce the influence of the noises from reseller. However, this leads to another problem that less data improves the possibility for collisions of other data items, which may trigger false positive response. A simple solution is for consumers to conduct verification process under all conditions (including buying from legal resellers), which can be used to trace the root for unauthorized reselling.

Acknowledgements

This work is supported in part by US National Science Foundation under grant numbers 1513719 and 1730291.

REFERENCES

[1] "Gumroad," https://gumroad.com/, [Online; accessed 17-Jul-2019].

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

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

[4] K. Poularakis, G. Iosifidis, I. Pefkianakis, L. Tassiulas, and M. May, "Mobile data offloading through caching in residential 802.11 wireless networks," *IEEE Transactions on Network and Service Management*, vol. 13, no. 1, pp. 71–84, 2016.

[5] N. Szabo, "Smart contracts: building blocks for digital markets," *EXTROPY: The Journal of Transhumanist Thought,(16)*, 1996.

[6] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[7] S. Diamond and S. Boyd, "Cvxpy: A python-embedded modeling language for convex optimization," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2909–2913, 2016.

[8] Microsoft Research, "Edge Computing," https://www.microsoft.com/en-us/research/project/edge-computing/, Oct. 2008.

[9] M. Satyanarayanan, V. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *IEEE pervasive Computing*, 2009.

[10] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, no. 5, pp. 2795–2808, 2016.

[11] D. Liu, B. Chen, C. Yang, and A. F. Molisch, "Caching at the wireless edge: design aspects, challenges, and future directions," *IEEE Communications Magazine*, vol. 54, no. 9, pp. 22–28, 2016.

[12] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, "Edge-centric computing: Vision and challenges," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 5, pp. 37–42, 2015.

[13] Y. Huang, F. Ye, and Y. Yang, "Peer data caching algorithms in large-scale high-mobility pervasive edge computing environments," in *2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC)*. IEEE, 2018, pp. 1–8.

[14] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[15] K. Fanning and D. P. Centers, "Blockchain and its coming impact on financial services," *Journal of Corporate Accounting & Finance*, vol. 27, no. 5, pp. 53–57, 2016.

[16] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, 2014.

[17] P. McCorry, S. F. Shahandashti, and F. Hao, "A smart contract for board-room voting with maximum voter privacy," in *International Conference on Financial Cryptography and Data Security*. Springer, 2017, pp. 357–375.

[18] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *Ieee Access*, vol. 4, pp. 2292–2303, 2016.

[19] D. K. Mulligan, J. Han, and A. J. Burstein, "How drm-based content delivery systems disrupt expectations of personal use," in *Proceedings of the 3rd ACM workshop on Digital rights management*. ACM, 2003, pp. 77–89.

[20] C. D'Orazio and K.-K. R. Choo, "An adversary model to evaluate drm protection of video contents on ios devices," *Computers & Security*, vol. 56, pp. 94–110, 2016.

[21] 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.

[22] X.-Y. Li, J. Qian, and X. Wang, "Can china lead the development of data trading and sharing markets?" *Communications of the ACM*, vol. 61, no. 11, pp. 50–51, 2018.

[23] T. Jung, X.-Y. Li, W. Huang, J. Qian, L. Chen, J. Han, J. Hou, and C. Su, "Accounttrade: Accountable protocols for big data trading against dishonest consumers," in *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*. IEEE, 2017, pp. 1–9.

[24] T. Chen, Y. Zhu, Z. Li, J. Chen, X. Li, X. Luo, X. Lin, and X. Zhang, "Understanding ethereum via graph analysis," in *INFOCOM 2018-IEEE Conference on Computer Communications, IEEE*. IEEE, 2018, pp. 1484–1492.