ARTICLE TEMPLATE

A Reward Response Game in the Blockchain-powered Federated Learning System

Suhan Jiang and Jie Wu

Department of Computer and Information Sciences, Temple University, USA

ARTICLE HISTORY
Compiled November 23, 2021

ABSTRACT
Mobile-crowd machine learning paradigm has been enabled due to the population of federated learning and the increasingly powerful mobile devices.This paper focuses on a mobile-crowd federated learning system that includes a central server and a set of mobile devices. The server, acting as a model requester, motivates all devices to train an accurate model by paying them based on their individual contributions. Each participating device needs to balance between the training rewards and costs for profit maximization. A Stackelberg game is proposed to model interactions between the server and devices. To match with reality, our model takes the training deadline and the device-side upload time into consideration. Based on different definitions of individual contribution, two reward policies, *i.e.*, the size-based policy and accuracy-based policy, are compared. The existence and uniqueness of Stackelberg equilibrium (SE) under both definitions are analyzed, according to which algorithms are proposed to achieve the corresponding SE(s). We show that there is a lower bound of 0.5 on the price of anarchy in the proposed game. We extend our model by considering the uncertainty in the upload time, where each device's upload time is subject to a normal distribution due to its unstable channel. We also utilize the blockchain technique to ensure a truthful, trust-free, and fair system. This paper also analyze how devices maximize their utilities when making profits via training as well as blockchain mining in the fixed-upload-time setting. A blockchain-powered testbed is implemented to reflect the presented federated learning system and experiments are conducted based on it to validate our analysis.

KEYWORDS
Blockchain technique; federated learning; incentive mechanism; mobile-crowd machine learning; price of anarchy; Stackelberg game

1. Introduction

Federated learning has enabled model(s) to be collaboratively trained across multiple devices using decentralized data samples without actual data exchange, and therefore protecting data privacy and security. Meanwhile, the growth of mobile devices also get machine learning at the end of the network for real. Therefore, mobile-crowd federated learning has emerged as a new business trend. Fig. 1 shows a typical federated learning system, consisting of a central server as a model requester and a set of mobile devices as model trainers. In such a system, the server distributes a global model to the devices. The devices train the model on locally available data. All updated models, instead of

CONTACT S.J. Author. Email: tug67249@temple.edu

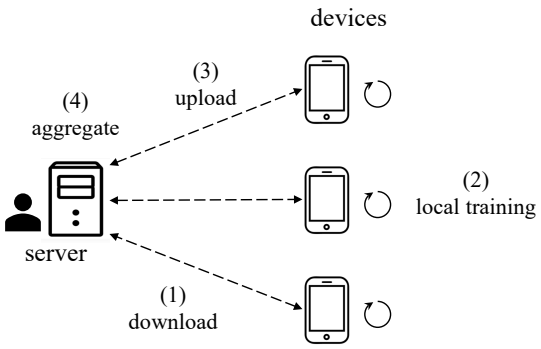


Figure 1.: Federated learning system: (1) server sends current global model to all devices; (2) each device trains its model using the local data; (3) all devices upload their updated models to server; (4) server aggregates all local models into a new global model.

the data, are then sent back to the server, where they are averaged to produce a new global model. This new model now acts as the primary model and is again distributed to the devices. This process is repeated forever or until the global model achieves a satisfactory result from the server side. Usually, the newly aggregated global model should get a little better than it already was. Obviously, model training moves to the edge of the network so that the data never leaves the device, while it is still under the central server's orchestration.

The fact that model training consumes resources makes it impossible for mobile devices to voluntarily participate in the federated learning task. In most cases, monetary incentive is a necessity for any mobile-crowd federated learning system. That is, the server has to motivate participating devices with enough rewards in order to obtain a satisfactory model. The model accuracy can be used to measure how satisfied the server is with the obtained model. Usually, the model accuracy is positively related to the size of overall trained data. Thus, the server wants devices to train more data in local training round, which inevitably will increase each device's cost. To cover their losses, more rewards should be provided. However, existing works also confirm that the model accuracy and the data size show a concave down increasing trend, indicating the principle of diminishing marginal return. Thus, to balance its utility, *i.e.*, the difference between its satisfaction of the obtained model and the reward it offers to all devices, it is important for a server to decide a suitable reward amount. In some case, the server may set a deadline by which all devices should complete the training and upload step, and ignore any update submitted after the deadline.

Devices participate in the federated learning aiming for the training rewards. Due to the different network environments, devices may vary over their own upload time. Each device must pay attention to its training time to avoid missing the deadline if one is given. Similarly, each device has its own computation power, indicating a specific training speed. The higher computation power a device has, the more data it can trains in a given time. Usually, a device will get rewarded based on its contribution to the global model. There exist two common ways to measure a device's individual contribution. One is using the size of its trained data, and the other is using the device's local model accuracy. As we mentioned before, the model accuracy is a concavely increasing function in terms of the training data size. Thus, the more rewards a device wants to obtain, the more data it has to train, and the more time it has to spend in the training step. However, a long training time also leads to a high computation cost. Obviously, the training time brings about a tradeoff between the reward and the

cost to the device. Thus, optimizing the training time is essential for each device, as it wants to maximize its utility, *i.e.*, the difference between its reward and its cost.

We exploit game theory to analyze the complex interactions between the server and mobile devices. To solve the reward-based resource management problem, we leverage a Stackelberg game, which includes two steps for the server (as a leader) and then devices (as followers), respectively. In the first step, the server announces its deadline and sets a reward for a training round by anticipating the devices' responses. In the second step, the devices decide their training time according to the observed reward and deadline as well as their individual upload times. Moreover, we investigate how the reward policy applied by the server will affect the devices' decisions as well as the whole system. As we mentioned before, a common reward policy can be paying each device either in proportion to its data-size-based contribution or to its local-accuracy-based contribution.

All previous studies assume that each device's upload time is fixed as a common knowledge in the proposed game. In practice, a mobile device in the wifi environment experiences an unstable network speed, indicating its upload time may change due to the time-varying network condition. In this paper, we also discuss the impact of upload time uncertainty on the devices' strategies by modeling each device's upload time as a random variable. That is, we assume that each device's upload time follows a Normal distribution with fixed values of mean and variance. To ensure that individual contributions are correctly measured and the promised reward is fairly distributed, we consider the blockchain technique to record all essential information for future verification. The importing of Blockchain allows devices to earn money via mining, which incurs a new challenge on how to split their computation power on training and mining for utility maximization. This paper also considers this regard in the fixed-upload-time setting.

The major contributions of this paper are as follows:

- We propose a Stackelberg game to solve a reward-based computation resource management problem in a federated learning system.
- We study the proposed Stackelberg game in two practical reward sharing policies, *i.e.*, size-based policy and accuracy-based policy, where the existence and uniqueness of Stackelberg equilibrium (SE) are analyzed.
- We show that our proposed game is a valid utility game, thereby its has a lower bound of 0.5 on the price of anarchy.
- We investigate the impacts of upload time uncertainty, which incurs longer training time on the device side.
- We utilize the blockchain technique to ensure a truthful contribution report and a fair reward allocation.
- We consider different roles devices can play in the proposed blockchain-based system and analyze their computation power splitting for utility maximization.
- We design a blockchain-powered testbed to implement the presented federated learning system and conduct experiments on it to validate our analysis.

The remainder of the paper is organized as follows. Section 2 presents our model and formulate the Stackelberg game. Section 3 analyzes equilibrium of the proposed game in the fixed-upload-time setting, by comparing two different reward policies. In Section 4, we investigate Price of Anarchy of the proposed game and find its lower bound. We further take the instability of communication channels between the server and devices scenarios into account and study new equilibrium in the variable-upload-

Table 1.: Summary of Notations.

Symbol	Description
R	total reward offered by the server in a global round
N	number of participating devices
β_i	unit-time training computation speed of device i
c_i	unit-time training computation cost of device i
α_i	device i's contribution
α / α_{-i}	sum of all devices' contributions with/without i
t_i	local training time decided by device i
τ_i	expected upload time of device i
T	server's pre-announced deadline
B_i	unit-time computation speed of device i
C_i	unit-time computation cost of device i
λ	training task arrival rate
M_p	total mining reward offered by Blockchain in the period of P

time setting in Section 6. Section 5 shows the utilization of the blockchain technique to ensure truthfulness and fairness, and we also consider different roles that devices can play in the proposed blockchain-based system and analyze their computation power splitting for utility maximization. We discuss simulation results in Section 7. Section 8 briefly gives the related backgrounds, and we conclude our paper in Section 9.

2. Model and Problem

2.1. Model Description

As shown in Fig. 1, we consider a cooperative federated learning system. The model consists of a number of mobile devices associated with a central parameter server. The whole system is in a universal mobile network with wireless communication infrastructures. We consider a quasi-static state where no devices are joining or leaving. Corresponding notations are shown in Table 1. The server aims to build a global machine learning model by employing N devices. Firstly, the server shares the current global model parameters with all devices. All devices will train their local models using their own data. Then, each device uploads its updated local model parameters to the server. Finally, the server facilitates the computation of the parameters aggregation and obtains a new global model. We consider that these four steps forms a global update round. Obviously, in such a global round, each device experiences lots of local training iterations, depending on its training data size. The global rounds continues repeatedly until meeting some specific requirements, *e.g.* a certain accuracy level or a deadline. Given that the model training and upload definitely bring about costs to the device side, the interaction between the server and all devices should be instantiated by an incentive mechanism.

Since the final global model is obtained through lots of training rounds, here, we only consider one round, in which, the server wants to make its model as improved as possible. According to the existing works, the accuracy of a machine learning model depends on the training data size. The relation between them can be captured by an

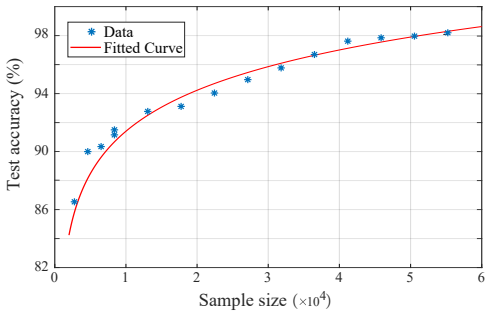


Figure 2.: Relation between the accuracy of the global model and the trained data size:$f(x) = 3.98log(9.68 \times 10^5 x - 3.69 \times 10^8)$.

concavely increasing function (an example is given in Fig. (2)), indicating a decreasing marginal gain. For simplicity, we assume that all training data in each edge device have the same quality and are independently and identically distributed (IID). Based on this assumption, the more data trained by the devices, the better global model the server will obtain at the end of a round. To motivate all devices' participation level, the server will announce a total reward R at the beginning of a round as an incentive. All devices will share this reward based on their individual contributions to the global model. Each device should train its local model and upload its local parameters to the server before the round deadline T. Here, we assume all devices simultaneously start local training at the time of 0. Let t_i and τ_i represent device i's local model training time and its parameter upload time. Then, the sum of t_i and τ_i cannot exceed the round deadline T. We define β_i as device i's unit-time computation speed, indicating that the number of its trained data is β_i in a unit time. Thus, in a training time t_i, device i trains $\beta_i t_i$ data in total. Let c_i represent the unit-time computation cost of device i. Then its total training cost will be $c_i t_i$. Obviously, a longer training time brings a higher data contribution ratio while also incurring more computation cost. Since all devices aim to make a profit, they should balance the contribution and the cost by carefully determining the training time t_i.

2.2. Stackelberg Game Formulation

We focus on incentive-driven interactions between the server and the devices. Each device's income varies according to the reward set by the server and all other devices' strategies, and its cost mainly depends on its individual strategy. In fact, the server decides its rewards by considering the devices' contributions. Game theory provides a natural paradigm to model the interactions between the server and the devices in this network. The server sets the total reward and announces it to the devices. The devices respond to the reward by deciding an optimal training time. Since the server acts first and then the devices make their decision based on the reward, the two events are sequential. Thus, we model the interactions between the server and the devices using a Stackelberg game. In our proposed game, the server is the leaders and the devices are the followers. It is a single-leader multi-follower Stackelberg game, the two stages of which can be described as follows.

In the first stage, the server optimizes the reward R it is willing to offer in a global round by predicting the devices' reactions. We assume the server also informs all devices of a deadline T, indicating that it only accepts models that arrived no later than the deadline. Devices that fail to successfully upload their local models will not

be rewarded. In the second stage, after observing the total reward and the deadline, each device i responds with a suitable training time, by taking its computation speed β_i, computation cost c_i, and upload time τ_i, as well as other devices' decisions into consideration. Since decisions are generated for individual utility maximization, a non-cooperative follower subgame is formed.

2.2.1. Device Side Utility

We define $\alpha_i(t_i)$ as device i's single round contribution. We will consider two different ways to define $\alpha_i(t_i)$, and under both definitions, the value of $\alpha_i(t_i)$ always depends on its training time t_i. In the rest of paper, we use α_i for the simplicity of writing. With the system model, we formulate the following optimization problem for maximizing the overall profit in each round:

Problem 1 ($\text{OP}_{\text{DEVICE}}$)**.**

$$\textit{maximize} \qquad u_i(t_i, \boldsymbol{t}_{-i}) = R\frac{\alpha_i}{\alpha} - c_i t_i, \tag{1a}$$

$$\textit{where} \qquad \alpha = \sum\nolimits_{j=1}^{N} \alpha_j, \tag{1b}$$

$$\textit{subject to} \qquad t_i + \tau_i \leq T. \tag{1c}$$

Each device i aims to maximize its utility and constraint (1c) ensures that i's local model can be uploaded within the deadline and thereby avoiding the worst case of zero-reward.

2.2.2. Server Side Utility

The objective of the server is to optimize its utility by determining the corresponding reward. Let V denote the server's utility, which is the difference between its satisfaction about the newly aggregated global model and the reward R it has to pay all qualified devices. We assume that the server's satisfaction is caught by the estimated accuracy of the new global model, which is a concavely increasing function over the amount of the data trained by all devices. Thus,, we use a log function to characterize the relationship between the model accuracy and the trained data. The optimization problem $\text{OP}_{\text{SERVER}}$ on the server side is thus defined as below.

Problem 2 ($\text{OP}_{\text{SERVER}}$)**.**

$$\textit{maximize} \qquad V = \theta \log\left(1 + \varepsilon \sum\nolimits_{i=1}^{N} \beta_i t_i\right) - R \tag{2}$$

2.2.3. Stackelberg Equilibrium

$\text{OP}_{\text{SERVER}}$ and $\text{OP}_{\text{DEVICE}}$ together form the proposed Stackelberg game. To achieve equilibrium in this game, where neither the leader (server) nor the followers (devices) have incentive to deviate, we need to find its subgame perfect Nash equilibrium (NE) in the follower stage first, and then apply backward induction to achieve the leader side equilibrium. Formally, the SE point(s) are defined as follows.

Definition 1. *Let* $\mathbf{t}^*$ *and* R^* *denote the optimal training time vector of all devices and the optimal reward offered by the server, respectively. Let* $\mathbf{t}^* = [t_1^*, \cdots, t_N^*]$*, then*

the point $(\mathbf{t}^*, R^*)$ *is the Stackelberg equilibrium if the following conditions hold:*

$$V(R^*, \mathbf{t}^*) \geq V_c(R, \mathbf{t}^*), \forall R, \tag{3a}$$
$$u_i(t_i^*, R^*) \geq u_i(t_i R^*), \forall t_i. \tag{3b}$$

3. Stackelberg Equilibrium in the Fixed-Upload-Time Setting

We start with a relatively simple setting, where each device has a stable channel connecting to the server. That is, the model upload time τ_i is a pre-known constant. This assumption allows us to focus on how different reward policies applied by the serve will affect devices' strategies and thereby influencing the result of the proposed Stackelberg game.

3.1. Size-based Reward Policy

3.1.1. Follower Subgame Equilibrium

It is natural to consider the size of the trained data to measure the individual contribution. The corresponding device side optimization problem can be rewritten as below.

Problem 3 ($\text{OP}_{\text{DEVICE}}$)**.**

$$\textit{maximize} \qquad u_i(t_i, \boldsymbol{t}_{-i}) = R\frac{\alpha_i}{\alpha} - c_i t_i, \tag{4a}$$
$$\textit{where} \qquad \alpha_i = \beta_i t_i, \tag{4b}$$
$$\textit{subject to} \qquad t_i + \tau_i \leq T. \tag{4c}$$

Theorem 1. *At least one Nash equilibrium exists in Problem 3.*

Proof. Obviously, each device's strategy space $[0, T - \tau_i]$ is a non-empty, compact, and convex subset of the Euclidean space, and the utility function $u_i(t_i, \mathbf{t}_{-i})$ is continuous and twice differentiable over $[0, T - \tau_i]$. In order to show the existence of Nash equilibrium, we need to prove that u_i is concave with respect to t_i. According to Eq. (5), the second order derivative of u_i is less than 0 over $[0, T - \tau_i]$.

$$\frac{\partial^2 u_i}{\partial t_i^2} = \frac{-2R\beta_i^2\alpha_{-i}}{\alpha^3} < 0 \tag{5}$$

where $\alpha_{-i} = \sum_{j \neq i} \alpha_j$. Therefore, we can conclude that there exists at least one Nash equilibrium in $\text{OP}_{\text{DEVICE}}$. □

Lemma 1. $\sqrt{\frac{R}{\beta_i c_i}} - \frac{2}{\beta_i}\sqrt{\alpha_{-i}} > 0$ *always holds if the following condition holds.*

$$2(N-1)\frac{c_i}{\beta_i} < \sum\nolimits_{j=1}^{N} \frac{c_j}{\beta_j} \tag{6}$$

Proof. Given the domain $[0, T - \tau_i]$ and the first order derivative Eq. (7) of u_i, device i's the best response strategy can be obtained in Eq. (8), which is a function over

$\mathbf{t} = [t_1, \cdots, t_N]$.

$$\frac{\partial u_i}{\partial t_i} = R\beta_i \frac{\alpha_{-i}}{\alpha^2} - c_i \tag{7}$$

$$t_i^* = g(\mathbf{t}) = \begin{cases} 0 & \frac{\sqrt{R\beta_i c_i \alpha_{-i}} - c_i \alpha_{-i}}{\beta_i c_i} < 0 \\ \frac{\sqrt{R\beta_i c_i \alpha_{-i}} - c_i \alpha_{-i}}{\beta_i c_i} & 0 < \frac{\sqrt{R\beta_i c_i \alpha_{-i}} - c_i \alpha_{-i}}{\beta_i c_i} \leq T - \tau_i \\ T - \tau_i & \frac{\sqrt{R\beta_i c_i \alpha_{-i}} - c_i \alpha_{-i}}{\beta_i c_i} > T - \tau_i \end{cases} \tag{8}$$

Let Eq. (7) equal to 0. Then, we obtain the following equation.

$$\frac{\alpha_{-i}}{\alpha^2} = \frac{1}{R} \frac{c_i}{\beta_i} \tag{9}$$

By summing up on the both sides of Eq. (9), we obtain $\sum_{j=1}^{N} \beta_j t_j = R(N-1)/\sum_{j=1}^{N} \frac{c_j}{\beta_j}$. According to Eq. (8), we obtain $c_i \left(\sum_{j=1}^{N} \beta_j t_j\right)^2 = R\beta_i \sum_{j \neq i} \alpha_j$. Combining these two equations, we obtain the following result.

$$\sum\nolimits_{j \neq i} \beta_j t_j = \left(\frac{N-1}{\sum_{j=1}^{N} \frac{c_j}{\beta_j}}\right)^2 \frac{R c_i}{\beta_i} \tag{10}$$

When Eq. (6) holds, we can easily prove that $\sqrt{\frac{R}{\beta_i c_i}} - \frac{2}{\beta_i} \sqrt{\sum_{j \neq i} \alpha_j} > 0$ always holds. □

Definition 2. *The function $g(\boldsymbol{t})$ is standard if for all $\boldsymbol{t} \geq 0$ the following properties are satisfied.*

(1) Positivity: $g(\boldsymbol{t}) > 0$,
(2) Monotonicity: if $\boldsymbol{t} \geq \boldsymbol{t'}$, then $g(\boldsymbol{t}) \geq g(\boldsymbol{t'})$.
(3) Scalability: $\forall \varepsilon > 1$, $\varepsilon g(\boldsymbol{t}) \geq g(\varepsilon \boldsymbol{t'})$.

Theorem 2. *There exists a unique Nash equilibrium in* $\mathrm{OP}_{\mathrm{DEVICE}}$ *if Eq. (6) holds.*

Proof. If Eq. (8) is a standard function, the proposed game has a unique Nash equilibrium. The positivity is obviously satisfied by $g(\mathbf{t})$. We prove the monotonicity of $g(\mathbf{t})$ under the condition Eq. (6) by showing $g(\mathbf{t}) - g(\mathbf{t}') \geq 0$ given $\mathbf{t} \geq \mathbf{t}'$.

$$\begin{aligned} g(\mathbf{t}) - g(\mathbf{t}') &= \sqrt{\frac{R}{\beta_i c_i}} \left(\sqrt{\sum_{j \neq i} \alpha_j} - \sqrt{\sum_{j \neq i} \alpha'_j} \right) - \frac{1}{\beta_i} \left(\sum_{j \neq i} \alpha_j - \sum_{j \neq i} \alpha'_j \right) \\ &= \left(\sqrt{\sum_{j \neq i} \alpha_j} - \sqrt{\sum_{j \neq i} \alpha'_j} \right) \left[\sqrt{\frac{R}{\beta_i c_i}} - \frac{1}{\beta_i} \left(\sqrt{\sum_{j \neq i} \alpha_j} + \sqrt{\sum_{j \neq i} \alpha'_j} \right) \right] \\ &\geq \left(\sqrt{\sum_{j \neq i} \alpha_j} - \sqrt{\sum_{j \neq i} \alpha'_j} \right) \left[\sqrt{\frac{R}{\beta_i c_i}} - \frac{2}{\beta_i} \sqrt{\sum_{j \neq i} \alpha_j} \right] \end{aligned} \tag{11}$$

Algorithm 1 Best Response Algorithm

Output: $\boldsymbol{t} = \{t_1, \cdots, t_N\}$
Input: Initialize k as 1 and pick a feasible starting point $\boldsymbol{t}^{(0)}$
1: **for** round k **do**
2: **for** device i **do**
3: Decide $t_i^{(k)} = t_i^{(k-1)} + \Delta \frac{\partial U_i\left(t_i, t_{-i}^{(k-1)}\right)}{\partial t_i}$
4: Send the local model the server
5: Server aggregates all models received before T into a new global model and send back to devices.
6: **if** $\boldsymbol{t}^{(k)} = \boldsymbol{t}^{(k-1)}$ **then** Stop
7: **else** set $k \leftarrow k + 1$

According to Lemma 1, the monotonicity property is proved.

To show scalability, we prove that $\forall \varepsilon > 1, \varepsilon g(\mathbf{t}) \geq g(\varepsilon \mathbf{t}')$ based on Eq. (12).

$$\varepsilon g(\mathbf{t}) - g(\varepsilon \mathbf{t}') = \varepsilon \sqrt{\frac{R \sum_{j \neq i} \alpha_j}{\beta_i c_i}} - \sqrt{\frac{R \sum_{j \neq i} \varepsilon \alpha_j'}{\beta_i c_i}} > 0 \tag{12}$$

Therefore, the proposed game always possesses a unique Nash equilibrium. □

This naturally gives a distributed iterative algorithm, allowing each device to iteratively update its strategy, given the strategies of other devices. We summarize the distributed iterative algorithm in Algorithm 1. Algorithm 1 is applicable to find the unique NE point, where each device is engaged in a gradient ascent process to maximize its utility.

Corollary 1. *When $T \to +\infty$, the unique Nash equilibrium for device i is given by*

$$t_i^* = \frac{R(N-1)}{\sum_{j=1}^{N} \frac{c_j}{\beta_j}} - \left(\frac{N-1}{\sum_{j=1}^{N} \frac{c_j}{\beta_j}} \right)^2 \frac{R c_i}{\beta_i}. \tag{13}$$

3.2. Accuracy-based Reward Policy

Another simple and intuitive way for the server to measure the individual contribution and distribute its reward is based on each device's local model accuracy. As we mentioned before, the relationship between the model accuracy and the training time can be characterized by a log function. As we will show in the below, when using model accuracy to measure device's contributions, α_i is still an strictly increasing concave function with respect to t_i. In this case, the new problem facing each device is shown as follows.

Problem 4 ($\mathrm{OP}_{\mathrm{DEVICE}}$)**.**

$$\textit{maximize} \qquad u_i(t_i, \boldsymbol{t}_{-i}) = R\frac{\alpha_i}{\alpha} - c_i t_i, \tag{14a}$$

$$\textit{where} \qquad \alpha_i = \theta \log\left(1 + \varepsilon \beta_i t_i\right), \tag{14b}$$

$$\textit{subject to} \qquad t_i + \tau_i \leq T, \quad \theta > 0, \quad \varepsilon > 0. \tag{14c}$$

Theorem 3. *At least one Nash equilibrium exists in Problem 7.*

Proof. The existence of Nash equilibrium can be confirmed by showing that its second

derivative is negative based on Eq. 15.

$$\frac{\partial^2 u_i}{\partial t_i^2} = -\frac{R\theta\varepsilon^2\beta_i^2\alpha_{-i}}{\left(1+\varepsilon\beta_i t_i\right)^2\alpha^2}\left(2\frac{\theta}{\alpha}+1\right) < 0 \tag{15}$$

Since the objective function is concave, we can conclude that there exists at least one Nash equilibrium in Problem 7. □

Lemma 2. *Let* $\Gamma = \left\{\mathbb{N}, (A_i)_{i\in\mathbb{N}}, (\pi_i)_{i\in\mathbb{N}}\right\}$ *be an N-player non-zero-sum game in normal form, where* $\mathbb{N}$ *represents the player set,* A_i *represents the i-th player's feasible strategies, which is a non-empty compact convex subset of the Euclidean space, and* π_i *represents the i-th player's utility function. If the utility functions* $(\pi_1, \cdots, \pi_N)$ *are diagonally strictly concave for* $(A_i)_{i\in\mathbb{N}}$*, then the game has a unique pure strategy Nash equilibrium [1].*

Lemma 3. *Given* $\bigtriangledown\pi(\mathbf{x}) = \left[\frac{\partial\pi_i}{\partial x_1}, \cdots, \frac{\partial\pi_N}{\partial x_N}\right]^\intercal$ *as the game's pseudo-gradient function and let* $\Pi(\mathbf{x})$ *denote the Jacobian of* $\bigtriangledown\pi(\mathbf{x})$*, if the symmetric matrix* $\Pi(\mathbf{x}) + \Pi^\intercal(\mathbf{x})$ *is negative definite for* $\mathbf{x} \in (A_i)_{i\in\mathbb{N}}$*, then the utility functions* $(\pi_1, \cdots, \pi_N)$ *are diagonally strictly concave for* $(A_i)_{i\in\mathbb{N}}$ *[1].*

Theorem 4. *There exists a unique Nash equilibrium in Problem 7.*

Proof. To prove the uniqueness of Nash equilibrium in Problem 7, we need to show that $U(\mathbf{t}) + U^\intercal(\mathbf{t})$, where U is given in Eq. (17), is negative definite.

We start with constructing the pseudo-gradient function $\bigtriangledown u(\mathbf{t}) = \left[\frac{\partial u_1}{\partial t_1}, \cdots, \frac{\partial u_N}{\partial t_N}\right]^\intercal$, where $\frac{\partial u_i}{\partial t_i}$ is shown in Eq. (16).

$$\frac{\partial u_i}{\partial t_i} = \frac{R\theta\varepsilon\beta_i\alpha_{-i}}{\left(1+\varepsilon\beta_i t_i\right)\alpha^2} - c_i, \quad \forall i \in [1, N] \tag{16}$$

Thus, we have the Jacobian of $\bigtriangledown u(\mathbf{x})$ as below.

$$U(\mathbf{t}) = \begin{bmatrix} \frac{\partial^2 u_1}{\partial t_1^2} & \frac{\partial^2 u_1}{\partial t_1 t_2} & \cdots \\ \frac{\partial^2 u_2}{\partial t_2 t_1} & \ddots & \\ \vdots & & \end{bmatrix} \tag{17}$$

where $\frac{\partial^2 u_i}{\partial t_i^2}$ is given in Eq. (15) and $\frac{\partial^2 u_i}{\partial t_i t_j}$ can be referred in Eq. (18).

$$\frac{\partial^2 u_i}{\partial t_i t_j} = \frac{R\theta^2\varepsilon^2\beta_i\beta_j\left[\alpha - 2(\alpha_i+\alpha_j)\right]}{\left(1+\varepsilon\beta_i t_i\right)\left(1+\varepsilon\beta_j t_j\right)\alpha^3} \tag{18}$$

Obviously, $U(\mathbf{t}) + U^\intercal(\mathbf{t})$ is symmetric. Due to the complexity of this matrix, we use Matlab to check its eigenvalues, which are all negative, indicating it is negative definite. □

Table 2.: Strategy iterations given $R = 1000$, $T = 250$, $N = 5$, $\theta = 10$, $\varepsilon = 2 \times 10^{-5}$ $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5) = (200, 200, 200, 200, 200)$, $(c_1, c_2, c_3, c_4, c_5) = (1.2, 1.4, 1.3, 1.4, 1)$, $(\tau_1, \tau_2, \tau_3, \tau_4, \tau_5) = (10, 10, 15, 75, 20)$.

			Round											
MEAS	t	INIT	1	2	3	4	5	6	7	8	9	10	11	Sum
Size	t_1	30	196	207	118	129	131	135	147	148	152	151	**151**	
	t_2	30	165	158	87	91	62	67	66	64	69	70	**71**	
	t_3	30	121	125	104	104	99	99	100	100	100	100	**100**	
	t_4	30	147	132	133	130	115	116	111	110	109	110	**110**	
	t_5	30	178	187	226	225	230	230	230	230	230	230	**230**	662
Accuracy	t_1	30	163	157	105	106	114	116	116	116	116	116	**116**	
	t_2	30	129	125	83	85	83	85	85	85	85	85	**85**	
	t_3	30	121	125	104	104	99	99	100	100	100	100	**100**	
	t_4	30	89	94	91	89	85	84	85	85	85	85	**85**	
	t_5	30	135	135	157	156	158	157	157	157	157	157	**157**	543

3.3. Comparison of Two Reward Policies

In Table 2, we give a five-device example to show the achieved Nash equilibrium under these two reward policies. Obviously, a device's local training time mainly depends on its cost-to-speed ratio, *i.e.*, c_i/β_i as well as its upload time when other outside factors are fixed. It is obvious that devices with lower cost-to-speed ratio and less upload time tend to have a longer training time under both reward policies. Further, we compare the difference of these two equilibria. We deliberately set identical initial values while we can still observe that the final equilibrium points are different. The size-based reward policy motivates devices to train for longer time as each second has the same value while the accuracy-based reward policy makes the later time less valuable, so that all devices tend to train for less time under this policy. The numerical results provided in Section 7 also shows the same result. Thus, we can say that the accuracy-based reward policy brings more benefit to the device-side.

3.4. Stackelberg Equilibrium

While Algorithm 1 achieves a unique pure strategy for the devices' game, our final goal is to obtain the Stackelberg equilibrium of the entire system. For this purpose, we leverage Algorithm 1 to construct the corresponding SE. The Stackelberg equilibrium of the game can be found by solving the following non-linear optimization problem. Let $\mathbf{t}^*(R)$ be the unique NE obtained by the followers when the server offers a reward of R. The server needs to solve the following optimization problem a priori to find its unique optimal reward R^* and announces it to the devices.

Problem 5 ($\mathrm{OP}_{\mathrm{SERVER}}$)**.**

$$\textit{maximize} \qquad V = \theta \log\left(1 + \varepsilon \sum\nolimits_{i=1}^{N} \beta_i t_i^*(R)\right) - R \tag{19}$$

The corresponding SE can be achieved by solving the Problem 5. We analyze the effects induced by these two different contribution definitions in the simulation part.

Specially, we compare the server utility, the total utility of all devices, and the social welfare achieved under these two definitions. Meanwhile, we also compare the social welfare computed under the proposed Stackelberg game with the optimal social welfare to see the price of anarchy caused by selfishness.

4. Robust Price of Anarchy

Consider that the federated learning system operates in a centralized control. That is, all devices follow the server's instruction to train their local models. Then the objective of this whole system should be maximizing the social welfare, denoted by W, *i.e.*, the difference between the global model accuracy and the training cost among all devices, which is given in Eq. (20).

$$W = \theta \log\left(1 + \varepsilon \sum\nolimits_{i=1}^{N} \beta_i t_i\right) - \sum\nolimits_{i=1}^{N} c_i t_i \tag{20}$$

Centralized control achieves a better performance than decentralized (game theoretic) control solutions in terms of the social objectives being met. The concept of price of anarchy, which is caused by the devices' selfish behaviors, is used to quantify the loss of efficiency in decentralized game solutions as compared to the optimal centralized control. In the following, we prove that the non-cooperative game played among all devices is a valid monotone utility game. As a result, we obtain a lower bound on the PoA of value 0.5.

Definition 3. *Let* $\Gamma = \left\{\mathbb{N}, (A_i)_{i\in\mathbb{N}}, (U_i)_{i\in\mathbb{N}}\right\}$ *be an N-player non-zero-sum game in normal form, where* $\mathbb{N}$ *represents the player set,* A_i *represents the i-th player's feasible strategies, which is a non-empty compact convex subset of the Euclidean space, and* U_i *represents the i-th player's utility function. Assume that the objective function* $W((A_i)_{i\in\mathbb{N}})$ *where* $w : 2^{(A_i)_{i\in\mathbb{N}}} \mapsto \mathbb{R}$*, is a general function defined over all subsets of* $(A_i)_{i\in\mathbb{N}}$*. Game* Γ *is called a valid utility game if it satisfies the following three properties*

(1) W *is submodular,*
(2) The objective value of a player is at-least his added value for the societal objective,
(3) The total value for the players is less than or equal to the total societal value.

And Γ *is called a monotone game if for all* $S \subseteq S' \subseteq (A_i)_{i\in\mathbb{N}}, W(S) \leq W(S')$*[20].*

Lemma 4. *If a game* Γ *is a valid monotone utility game, then its lower bound on the PoA is* 0.5.

Theorem 5. *Our proposed Stackelberg game has a lower bound on the PoA of* 0.5.

Proof. We show that our proposed game has the following three properties:

(1) W is submodular.
Assume there exists one set $\mathbf{a} \subseteq \mathbf{t}$ and two elements $t_p, t_q \in \mathbf{t} - \mathbf{a}$. We define set $\mathbf{a}' = \mathbf{a} \cup \{t_p\}$, indicating that $\mathbf{a} \subseteq \mathbf{a}' \subseteq \mathbf{t}$. Therefore, we have

$$\left(W(\mathbf{a} \cup \{t_q\}) - W(\mathbf{a})\right) - \left(W(\mathbf{a}' \cup \{t_q\}) - W(\mathbf{a}')\right)$$

$$
\begin{aligned}
=&\theta \log\left(\frac{1+s+\varepsilon\beta_q t_q}{1+s}\right) - \theta \log\left(\frac{1+s+\varepsilon\beta_p t_p+\varepsilon\beta_q t_q}{1+s+\varepsilon\beta_p t_p}\right)\\
=&\theta \log\left(\frac{(1+s+\varepsilon\beta_q t_q)(1+s+\varepsilon\beta_p t_p)}{(1+s)(1+s+\varepsilon\beta_p t_p+\varepsilon\beta_q t_q)}\right) > 0,
\end{aligned} \tag{21}
$$

where $s = \varepsilon \sum_{i\in\mathbf{a}} \beta_i t_i$. Thus, we can conclude that W is a submodular function.

(2) Device i's utility u_i is at-least its added value for the societal objective.
To prove this property, we need to show that $u_i(t_i, \mathbf{t}_{-i}) \geq W(t_i, \mathbf{t}_{-i}) - W(\mathbf{t}_{-i})$.

$$
\begin{aligned}
&u_i(t_i,\mathbf{t}_{-i}) - (W(t_i,\mathbf{t}_{-i}) - W(\mathbf{t}_{-i}))\\
=&R\frac{\alpha_i}{\alpha} - \theta \log\left(\frac{1+\varepsilon\sum_{j=1}^{N}\beta_j t_j}{1+\varepsilon\sum_{j=1}^{N}\beta_j t_j - \varepsilon\beta_i t_i}\right)\\
=&R\frac{\alpha_i}{\alpha} - \theta \log\left(1+\frac{\varepsilon\beta_i t_i}{1+\varepsilon\sum_{j=1}^{N}\beta_j t_j - \varepsilon\beta_i t_i}\right) > 0
\end{aligned} \tag{22}
$$

(3) The total value for the devices is less than or equal to the total societal value.
Here, we show the difference of $\sum_{i=1}^{N} u_i$ and W as below.

$$
\begin{aligned}
&\sum_{i=1}^{N} u_i - W\\
=&\sum_{i=1}^{N}\left(R\frac{\alpha_i}{\alpha} - c_i t_i\right) - \left(\theta \log\left(1+\varepsilon\sum_{i=1}^{N}\beta_i t_i\right) - \sum_{i=1}^{N} c_i t_i\right)\\
=&R - \theta \log\left(1+\varepsilon\sum_{i=1}^{N}\beta_i t_i\right) > 0
\end{aligned} \tag{23}
$$

Now, we can conclude that our proposed game is a valid monotone utility game. Thus, its has a lower bound of 0.5 on the PoA. □

5. Unstable Communication Chanel

As we mentioned before, local updates have to be transferred to the server. Every such update is of the same size as the trained model, which can be in the range of gigabytes for modern architectures with millions of parameters [2,3]. Nevertheless, the devices typically employed in federated learning are communication-constrained, for example IoT devices or smartphones are generally connected to Wifi networks. Obviously, our previous assumption that each device has a fixed upload time τ_i is not realistic due to the mobility of devices and instability of wifi connection. In the following, we further consider a complex setting, where each device i's upload time is stochastic and subject to a normal distribution $\mathcal{N}(\mu_i, \sigma_i^2)$ for $\forall i \in [1, N]$. In the following, we will focus on equilibrium analysis in the size-based-policy setting, while it also holds in the accuracy-based-policy setting.

5.1. Problem Formulation

Since device i's upload time follows a normal distribution $\mathcal{N}(\mu_i, \sigma_i^2)$, the probability that i successfully uploads its model within time τ_i can be expressed as Eq. (24).

$$F_i(\tau_i) = \int_{-\infty}^{\tau_i} \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left\{-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right\} \mathrm{d}x \tag{24}$$

After a device i decides on its training time t_i, it has a time period of $T - t_i$ for uploading. Therefore, its model can be successfully uploaded with the probability of $F_i(T - t_i)$. Any update after the deadline T will not be accepted by the server. Thus, with a probability of $F_i(T - t_i)$, device i can contribute a set of data $\alpha_i = \beta_i t_i$ to the global model, otherwise, its data contribution will be 0. Since all other devices follow the same principle to participate in this game as well, device i can estimate that the total data contributed by other devices would be $\sum_{j\neq i} \beta_j t_j F_j(T - t_j)$ in expectation, which we denote as $\hat{\alpha}_{-i}$ for simplification. Thus, if device i successfully uploads its model, then the system-wide data contribution is $\hat{\alpha}_{-i} + \beta_i t_i$. Based on the analysis above, we reformulate the optimization problem for an individual device i.

Problem 6 ($\mathrm{OP}_{\mathrm{DEVICE}}$)**.**

$$\textit{maximize} \qquad u_i(t_i, \boldsymbol{t}_{-i}) = R\frac{\alpha_i F_i(T-t_i)}{\hat{\alpha}_{-i} + \alpha_i} - c_i t_i, \tag{25a}$$

$$\textit{subject to} \qquad 0 \leq t_i < T. \tag{25b}$$

In fact, each device's utility function fails to satisfy the quasi-concavity condition in the strategy space. However, a non-concave game still possesses a pure strategy Nash equilibrium when meeting some specific conditions which is given in Lemma 5 [4].

Lemma 5. *Let* $\Gamma = \left\{\mathbb{N}, (A_i)_{i\in\mathbb{N}}, (U_i)_{i\in\mathbb{N}}\right\}$ *be an N-player non-zero-sum game in normal form, where* $\mathbb{N}$ *represents the player set,* A_i *represents the i-th player's feasible strategies, which is a non-empty compact convex subset of the Euclidean space, and* U_i *represents the i-th player's utility function. Assume that for each* $i \in \mathbb{N}$*:*

(1) A_i *is some closed interval of the real line,*
(2) $U_i(.)$ *is continuous on* A_i*,*
(3) For each $\boldsymbol{x}_{-i} \in \boldsymbol{A}_{-i}$*, there exists a local maximum of* $U_i(\boldsymbol{x}_{-i}, .)$*, and this local maximum is also a global maximum,*

then the game Γ *possesses a pure-strategy Nash equilibrium.*

Theorem 6. *Nash equilibrium exists in* $\mathrm{OP}_{\mathrm{DEVICE}}$*.*

Proof. Obviously, our proposed game satisfies the conditions (1) and (2). Thus, we now prove that $\forall \mathbf{x}_{-i}$, $u_i(t_i, \mathbf{t}_{-i})$ has a local maximum over its strategy domain $[0, T)$ and that this local maximum is also a global maximum of $u_i(t_i, \mathbf{t}_{-i})$ over its feasible domain.

For each feasible $\mathbf{t}_{-i}$, we start to analyze the function $u_i(t_i, \mathbf{t}_{-i})$'s monotonicity with respect to t_i. We show the first-order derivative of u_i in the below

$$\frac{\partial u_i}{\partial t_i} = R\frac{\beta_i F_i(T-t_i)\hat{\alpha}_{-i} - \beta_i t_i f_i(T-t_i)\left(\hat{\alpha}_{-i} + \beta_i t_i\right)}{\left(\hat{\alpha}_{-i} + \beta_i t_i\right)^2} - c_i$$

T \ R	200	400	600	800	1000
100	32	64	90	90	90
120	32	64	96	110	110
160	32	64	96	128	150

(a) Size-based policy.

T \ R	200	400	600	800	1000
100	31	61	90	90	90
120	31	61	90	110	110
160	31	61	90	118	144

(b) Accuracy-based policy ($\theta=10, \varepsilon=8\times10^{-6}$).

T \ R	200	400	600	800	1000
100	29	54	76	90	90
120	29	54	76	97	110
160	29	54	76	97	117

(c) Accuracy-based policy ($\theta=10, \varepsilon=4\times10^{-5}$).

Table 3.: Homogeneous follower subgame Nash equilibrium under different rewards and deadlines where $(N, \beta, c, \tau) = (5, 200, 1, 10)$.

Obviously, $\frac{\partial u_i}{\partial t_i}$ is continuous over its feasible domain. We show the signs of $\frac{\partial u_i}{\partial t_i}$ on the boundary points over its strategy space.

$$\frac{\partial u_i}{\partial t_i}(0, \mathbf{t}_{-i}) = \frac{R\beta_i F_i(T)}{\hat{\alpha}_{-i}} - c_i \tag{26}$$

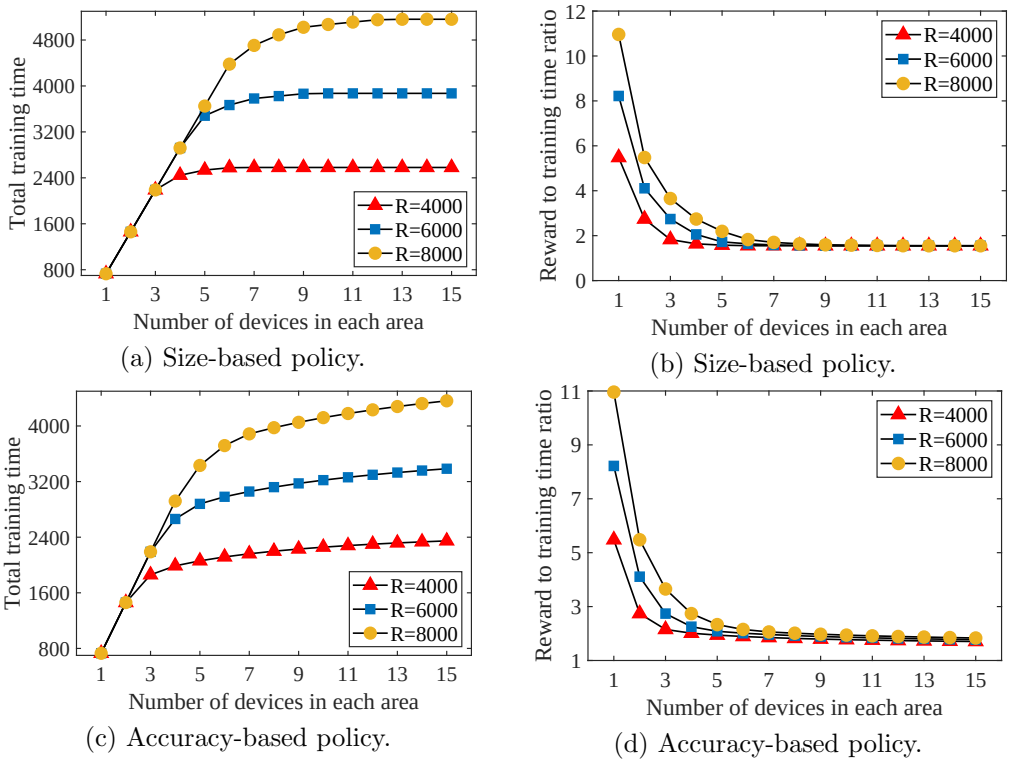
$$\frac{\partial u_i}{\partial t_i}(T, \mathbf{t}_{-i}) = R\frac{\beta_i F_i(0) - \beta_i t_i f_i(0)\,(\hat{\alpha}_{-i} + \beta_i T)}{(\hat{\alpha}_{-i} + \beta_i T)^2} - c_i \tag{27}$$

Since Eq. (26) is positive and Eq. (27) is negative, there must exist a certain $t_i^* \in (0, T)$ so that $\frac{\partial u_i}{\partial t_i}(t^*, \mathbf{t}_{-i}) = 0$, and u_i increases in the domain $[0, t_i^*)$ and decreases in the domain $(t^*, T]$. Thus, u_i reaches its local maximum at the point t_i^*.

Next, we will prove that $u_i(t_i^*)$ is a global maximum in its feasible domain $[0, +\infty)$. When $t_i >= T$, the value of $F_i(t_i) = 0$ always holds, meaning that $\frac{\partial u_i}{\partial t_i}(t_i, \mathbf{t}_{-i}) < 0$ holds, thus u_i is a decreasing function in terms of t_i in the domain of $[T, +\infty)$. Therefore, $u_i(t_i^*)$ is a global maximum as well. Now, we can conclude that the proposed game possesses a pure-strategy Nash equilibrium. □

6. Blockchain-powered System

The previous discussions are based on a strong assumption that the server and devices are honest and reliable. That is, each device truthfully reports its contribution, *i.e.*, how much data it has trained or how accurate its model is, and the server fairly distributes the reward as promised among all devices based on their contributions. However, in reality, a device may exaggerate its contribution for more reward and the server may refuse to make a payment after receiving models from devices. To keep the whole system operating orderly, we utilize the blockchain technique [5] and a



(a) Size-based policy. (b) Size-based policy.

(c) Accuracy-based policy. (d) Accuracy-based policy.

Figure 3.: Impact of device number under $T = 160$.

smart contract [6] to fulfill these two requirements, *i.e.*, device-side contributions are quantified correctly and the server-side payment is implement as promised.

6.1. Server-Device Smart Contract Workflow

Before the system operates, the server and all participating devices are required to sign a smart contract, which will be recorded in the blockchain for future query and validation. The workflow of such a smart contract is as below.

In the beginning of a round, the server sends its training request to the smart contract by announcing its reward R and deadline T. Meanwhile, it has to send an amount of R deposit to the smart contract. After the deposit is ready, the smart contract will retrieve the current global model and send it together with the reward and the deadline information to all participating devices. All devices utilize received information to determine their strategies and then train their local models. All the newest local models are uploaded to the smart contract. The smart contract will automatically record each model's arriving time so that all models arriving after the deadline will not be accepted. When reaching the deadline, the smart contract will aggregate all available local models into a new global model. To measure each device's contribution, there is a measurement function with a testing dataset to estimate each device's contribution. All devices' contributions are determined using the same measurement function, meaning that the measuring criteria is uniform and fair. Based on the estimated contribution, the smart contact will distribute the deposit of R to each device.

6.2. Device Computation Power Splitting

Since we import Blockchain, it is natural to consider about block mining, which maintains and secures our system's operation. Mining itself is profitable in the long term and consumes computation power. Except training, devices can also make profits in our Blockchain-based system by contributing its computation power to mining. Here, we assume all devices also work as Blockchain mining nodes, and hence are responsible for both training and mining. We assume that, devices are mining all the time. When training tasks come, they will perform training and mining. That is, for a device, it will split its computation power to training and mining when taking both tasks. Otherwise, all of its computation power will be devoted to mining only.

In the current setting where devices can perform training as well as mining, we assume that device i's total computation power can lead to a unit-time speed of B_i, at which i can mine without power splitting. After splitting, i's unit-time training speed becomes β_i and its unit-time mining speed decrease to $B_i - \beta_i$. Previously, β_i is assumed to be known. Now, it becomes a variable that should be determined by device i to optimize its overall utility on training and mining. Let's assume the arrival of training tasks follow a Poisson process. That is, on average, in each unit time, there are λ tasks required by the server. We simply assume all tasks are homogeneous so that they should have the same value of R and T. And each device has a fixed upload time. In our setting, both λ and T are dedicatedly determined to ensure that an old task has been completed before a new one comes. All devices care about their own utility in a period of time, say P. That is, each device determine its strategies to maximize its totally utility in the period of P, during which, the total mining reward offered by Blockchain is supposed to be M_p.

Since device i's training time for each task is t_i, its total training time will be Pt_i/λ, meaning that its full mining time will be $P - Pt_i/\lambda$ in total. Mining reward is distributed among all mining nodes based on their computation power contribution in a proportional way. Thus, device i's mining reward can be captured as below:

$$M_p \frac{PB_i - P\frac{t_i\beta_i}{\lambda}}{\sum_j \left(PB_j - P\frac{t_j\beta_j}{\lambda}\right)} = M_p \frac{\lambda B_i - t_i\beta_i}{\lambda \sum_j B_j - \sum_j t_j\beta_j}. \tag{28}$$

And its training reward should be $(P/\lambda) \cdot (R\alpha_i/\alpha)$. Let C_i be the unit-time computation cost for device i. Then its total computation cost is C_iP, Now, we are ready to reformulate the utility function for each device.

Problem 7 ($\mathrm{OP}_{\mathrm{DEVICE}}$)**.**

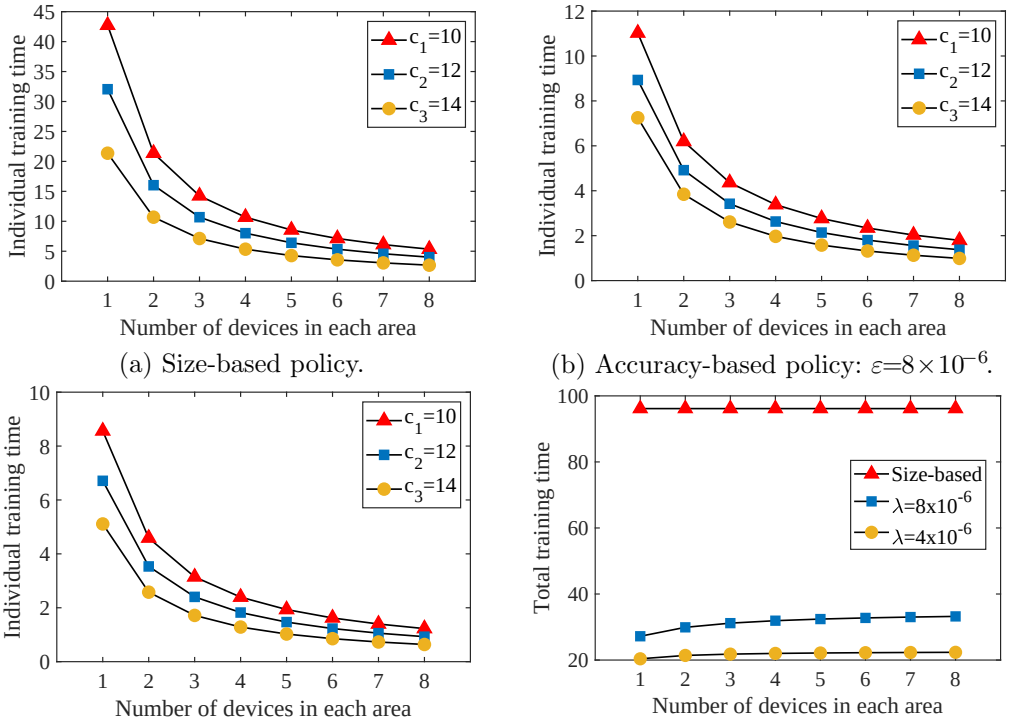
$$\begin{aligned} \textit{maximize} \quad & U_i(\beta_i, \beta_{-i}, t_i, \boldsymbol{t}_{-i}) \\ & = M_p \frac{\lambda B_i - t_i\beta_i}{\lambda \sum_j B_j - \sum_j t_j\beta_j} + R\frac{\alpha_i P}{\alpha\lambda} - C_iP, && \text{(29a)} \\ \textit{subject to} \quad & t_i + \tau_i \le T, \quad \theta > 0, \quad \varepsilon > 0. && \text{(29b)} \end{aligned}$$

U_i's concavity is determined based on Eq. (28), given RP/λ and C_iP are constant and α_i/α is concave. Since Eq. (28) is concave over t_i and β_i, we can conclude the existence of Nash equilibrium in the current setting.

7. Simulation

Our evaluation includes four parts. First, we examine how the server and the devices (Subsection VII.A and B) decide their optimal strategies. Second, we compare the game-driven market equilibrium and the optimal social welfare to confirm our PoA lower bound (Subsection VII.C). Then, we analyze how the upload channel jitters influence the achieved equilibrium (Subsection VII.D). Lastly, we show how devices reach their maximal utilities when splitting their computation power on training and mining. All experiment data confirm our theoretical results and show the efficiency of our proposed algorithm.

We conduct our experiments using Tensorflow 1.9 (to get fine-tuned machine-learning related parameters) and Matlab R2019b (to help the server and devices make decisions) on Ubuntu 16.04 LTS. Our smart contract (to build a trustless trade) is implemented through CITA [7], a blockchain framework that supports smart contract design and execution.



(a) Size-based policy.

(b) Accuracy-based policy: ε=8×10^{-6}.

(c) Accuracy-based policy: ε=4×10^{-6}.

(d) Size vs Accuracy.

Figure 4.: Impact of device number under $T = 80$.

7.1. Follower Subgame Nash Equilibrium

In this part, we will first analyze Nash equilibrium achieved among all devices. We will discuss how different parameters will affect the devices' equilibrium strategies.

7.1.1. Parameters from the server side

The server can determine its deadline T, its reward R, and its reward policy, *i.e.*, size-based or accuracy-based. In the following, we focus on investigating how those

t R	200	400	600	800	1000
t_1	45.8	91.5	110	110	110
t_2	29.1	58.3	91.4	110	110
t_3	29.1	58.3	91.4	95	95
t_4	12.5	25	43.4	80	110
t_5	12.5	25	43.4	80	105
sum	**129**	**258.1**	**379.6**	**475**	**530**

(a) $T = 120$.

t R	200	400	600	800	1000
t_1	45.8	91.5	137	150	150
t_2	29.1	58.3	87.4	121	150
t_3	29.1	58.3	87.4	121	135
t_4	12.5	25	37.5	57.1	88.1
t_5	12.5	25	37.5	57.1	88.1
sum	**129**	**258.1**	**386.8**	**506.2**	**611.2**

(b) $T = 160$

Table 4.: Heterogeneous follower subgame Nash equilibrium under size-based policy.

parameters affect the device-side equilibrium. We start with a simple homogeneous-device setting, where $(N, \beta, c, \tau) = (5, 200, 1, 10)$. In Table 3, we show the impact caused by different decisions on deadline T, the reward R, and the reward policy. Note that, R may not be its equilibrium value. Obviously, the increase of R's value is the main driven power for all devices to extend their training time. In the size-based reward policy, we can even observe the linear relation between the reward and the training time in some case. However, after devices reach their optimal training time, the server cannot push them to train longer by providing more rewards. This indicates that, the server should carefully determines its reward value to avoid useless monetary invest. This is an important reason why we utilize Stackelberg model since it adds the leader level to ensure the server's benefit.

Based on Table 3, we can also confirm the importance of the deadline T as it is the upper bound of training time. For example, in Table 3(a), given $R = 1000$, the optimal training time can be $t = 160$ if without considering the deadline, meaning that devices are willing to train more time and the server can obtain a better model. However, the deadline prevents devices from reaching their optimal training time and lower the server's utility. Although, in our theoretical discussion, we assume T's value is a pre-announced constant, it also plays an essential role for both the server and all devices. It can be our future work by adding the deadline T as another variable to be optimized by the server in our proposed Stackelberg game.

By comparing Table 3(a) to Table 3(b) and Table 3(c), we can conclude that, in most cases, the size-based reward policy leads devices to train for a longer time if other parameter values are identical, and thus, bringing more benefit to the server. By comparing Table 3(b) and Table 3(c), we also see the influence caused by the accuracy measurement function. Generally, an accuracy measurement function with the higher diminishing return will motivate devices for a longer time training. Note that, the accuracy measurement functions in Table 3(b) and Table 3(c) are transformed based on the results obtained by training the open datasets Reddit and Celeba, respectively.

t\R	200	400	600	800	1000
t_1	36.7	64.9	89.6	110	110
t_2	25.5	47.2	66.9	85.5	106
t_3	25.5	47.2	66.9	85.5	95
t_4	16.1	33.1	49.3	64.8	82.5
t_5	16.1	33.1	49.3	64.8	82.5
sum	**119.9**	**225.5**	**322**	**410.6**	**476**

(a) $T = 120$.

t\R	200	400	600	800	1000
t_1	36.7	64.9	89.6	112	133
t_2	25.5	47.2	66.9	85.3	102.6
t_3	25.5	47.2	66.9	85.3	102.6
t_4	16.1	33.1	49.3	64.6	79.2
t_5	16.1	33.1	49.3	64.6	79.2
sum	**119.9**	**225.5**	**322**	**411.8**	**496.6**

(b) $T = 160$

Table 5.: Heterogeneous follower subgame Nash equilibrium under accuracy-based policy($\theta = 10, \varepsilon = 4 \times 10^{-5}$).

7.1.2. Number of Participating Devices

In this part, we investigate the impact caused by the number of participating devices N. We assume all devices are evenly distributed in 5 areas. All devices have the same computation speed and devices located in the same area enjoy the identical unit cost and upload time. The detailed setting are given as $T = 160$, $\beta = 200$, $(c_1, c_2, c_3, c_4, c_5) = (1, 1.2, 1.2, 1.4, 1.4)$, and $(\tau_1, \tau_2, \tau_3, \tau_4, \tau_5) = (10, 10, 25, 10, 15)$. We change the number of devices in each area so that the total device number ranges from 5 to 75 and we show the device number impact in Fig. 3. According to Fig. 3(a) and Fig. 3(c), we can conclude that, in the beginning, the increase on the device number can result in a longer total training time, *i.e.*, the sum of all devices' training time. When reaching some point, the increasing trend stops, meaning that the newly joining devices only splits the reward with the existing devices while bringing no benefits to the system. As we can see in Fig. 3(b) and Fig. 3(d), the ratio between the reward R and the total training time converges to the same value in the end. Thus, blindly recruiting more devices cannot increase the global model's accuracy while brings more ordination work to the server.

7.1.3. Device Parameters

Now we study how the values of (β, c, τ) affect individual devices' equilibrium strategies. Here, we still apply the five-area setting we mentioned in the above, while we assume there is only one device in each area, *i.e.*, $N = 5$. We find each device's equilibrium strategy under different system parameters. Table 4 and Table 5 show the results under the size-based reward policy and the accuracy-based reward policy, respectively. Based on these two tables, we can see the training time heavily depends on a device's computation cost to speed ratio when other conditions are identical. Meanwhile, its upload time also constrains its training length as no device wants to miss the deadline.

We can conclude that devices with lower cost-to-speed ratio and less upload time tend to have a longer training time under any reward policy.

σ \ R	100	200	300	400	800
0	16	32	48	64	125
1	23	34.5	48.4	64	128
10	27	38	50	64.1	128

(a) Size-based policy.

σ \ R	100	200	300	400	800
0	15.8	31.2	46.3	61.1	117.6
1	17.8	31.3	46.3	61.1	117.6
10	24.6	35.9	46.8	61.1	117.6

(b) Accuracy-based policy ($\theta = 10, \varepsilon = 8\times10^{-6}$).

σ \ R	100	200	300	400	800
0	15.1	28.8	41.7	53.8	97.3
1	17.7	28.9	41.7	53.8	97.3
10	23.9	34.5	43.1	53.9	97.3

(c) Accuracy-based policy ($\theta = 10, \varepsilon = 4\times10^{-5}$).

Table 6.: Homogeneous follower subgame Nash equilibrium under different rewards and deadlines where $(N, T, \beta, c, \tau) = (5, 140, 200, 1, 15)$.

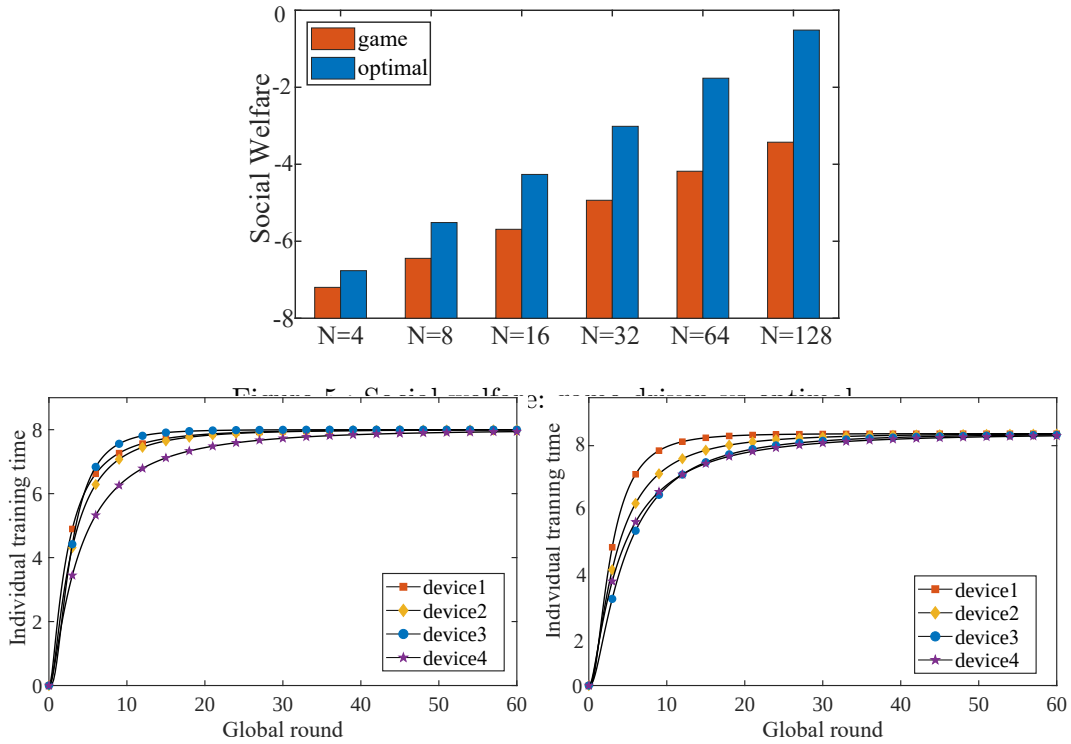
7.2. Leader-Follower Stackelberg Equilibrium

Based on the device-side analysis, we further study the optimal strategy on the server side to obtain the desired Stackelberg equilibrium.We consider a three-area setting and the detailed setting are given as $T = 80$, $\beta = 10000$, $\tau = 15$, and$(c_1, c_2, c_3) = (10, 12, 14)$.

We investigate the Stackelberg equilibrium under different reward policy. We show the device-side equilibrium strategies in Fig. 4. And we can find that the total training time in each area is almost fixed when increasing the device number in these areas. We find another interesting observation that, although we change the number of devices in each area, the server's optimal reward is almost the same. The server's optimal reward is around 1731, 626, and 408, for Figs. 4(a), (b), and (c) respectively. According to Fig. 4(d), we can observe the devices' total training time is positively related to the server's reward, which matches with the real market.

7.3. Price of Anarchy

In this section, we want to compare the social welfare created by different model designs.The definition of social welfare is the difference between the global model satisfaction and the total cost on the device side. We focus on the optimal control model and our proposed Stackelberg game. The optimal control means no money incentive: all devices are forced to follow the central server's scheduling. We consider a homogeneous-device setting. We show the obtained social welfare of these two models under different device numbers in Fig. 5. We can see that the social welfare yielded by our proposed Stackelberg game is always more than half of the optimal social welfare



(a) P contains 3 mining rounds. (b) P contains 5 mining rounds.

Figure 6.: Different lengths of P.

under a central controller, which confirms our theoretical analysis on the lower bound of PoA.

7.4. Uncertainty in Upload Time

In this part, we investigate the impact caused by the channel instability. We still apply the homogeneous-device setting, where $(N, \beta, c, \tau) = (5, 200, 1, 10)$. We use different values of σ to reflect how unstable the communication channels are. Note that, $\sigma = 0$ is the special case, representing that the upload time is fixed as 10. According to Table 6, we can conclude that, the uncertain upload time makes devices spend more time on the local training.

7.5. Device Computation Power Splitting

We perform our experiment in the 4-identical-device setting. We start with the setting that P contains 3 or 5 mining rounds. As is shown in Fig. 6(a), all devices' strategies finally converge to the same point after 50 mining rounds, since they are identical. This result confirms the existence of Nash equilibrium in the device computation power splitting game.

To show the impact caused by the length of the assessment period, we extend P to contains 5 mining rounds. We show each device's strategy in Fig. 6(b). We find that in this setting, devices spend more time on training and the time used for convergence becomes shorter.

8. Related Work

8.1. Federated Learning

As there is more and more attention on privacy, federated learning has become one of the essential concepts in modern machine learning. Existing works in this research field can be divided into two directions. In one direction, researchers focus on solving the global model accuracy decreasing caused by device heterogeneity [8–10] in terms of hardware, network connectivity, and battery power, and data heterogeneity among all devices [11–13]. This paper focuses on how to improve the global model accuracy by motivating all participating devices train more data locally. Some literature also consider to dealing with the coordination and operation problems in such a system, such as the communication bottleneck [14–17] and the trust and truthfulness [18] between the server and devices. Like [19,20], our paper also utilizes the blockchain and a smart contract to ensure that the devices' contributions are correctly quantified and the reward promised by the server are fairly distributed.

8.2. Incentive Mechanism Design for Mobile Crowdsourcing

An effective incentive mechanism is indispensable in mobile crowdsourcing tasks. Devices need to consume resources to perform tasks and the crowdsourcer needs to adjust the rewards to devices based on the task difficulties and device performances. Most solutions integrate online auction and game theory techniques for mechanism design [21–23]. There also exist some works dealing with incentive mechanisms in the federated learning system. In [24], the authors present an incentive mechanism called FMore with multi-dimensional procurement auction to select high-quality training nodes. [25] utilizes contract theory in order to motivate high-reputation workers to join model training. Our paper utilizes a Stackelberg game, where the server acts as a leader and provides rewards based on devices' individual contributions to motivate each device to feed its local model with more data in each iteration. We utilize utility theory, which has been widely applied to decision making [26,27], and design suitable utility functions for both the server and the devices. Besides, our model is more practical as we take the training deadline and device upload time into consideration.

8.3. Price of Anarchy

In algorithmic game theory, the price of anarchy (PoA) [28] is defined as the ratio of the social cost of a worst Nash equilibrium to that of a social optimum (*i.e.*, an assignment of strategies to players achieving optimal social cost). This highly successful and influential concept is frequently thought of as the standard measure of the potential efficiency loss due to individual selfishness, when players are concerned only with their own utility and not with the overall social welfare. Lots of works dealing with communication network problems either use this concept to measure the efficiency their methods can achieve [29–33] or utilize this concept as the system design goal [34,35]. In this paper, we investigate the existence of a pure strategy equilibrium in a resource management game and measure the inefficiency of equilibria by the price of anarchy. We show that the lower bound on the price of anarchy is 0.5 for our proposed solution.

8.4. Blockchain-based FL Systems

The combination of blockchain and FL promises the secure data sharing method in the edge of the network and the main advantage of integrating blockchain into FL systems is to provide a trust-free and fair environment for the training of decentralized models. The blockchain can store all those trained parameters in a secure and immutable way with the resistance against unauthorized access and malicious actions. Most of the existing Blockchain-based FL Systems focus on protecting privacy, achieving decentralization and improving the performance of model training [36–39], while our paper pays attention to computation power allocation on mining and training simultaneously by considering different roles that devices can play.

9. Conclusion

In this paper, we utilize a Stackelberg game to model the interaction between a server and all participating devices in a federated learning system. We aims to find the server's optimal reward the server and each device's optimal training time for the purpose of individual utility maximization. Our model takes both the server-side deadline and the device-side upload time into consideration. We consider two different reward policies, *i.e.*, size-based and accuracy-based, and investigate how they affect the equilibrium achieved in the whole system. We prove that the proposed game is a valid utility game, which has a lower bound of 0.5 on the PoA. We also extend our model by adding uncertainty in the upload time. We show that devices spend more time on local training in the variable-upload-time setting. We design a blockchain-powered testbed to implement the presented federated learning system and allow devices to execute mining and training simultaneously. Experiments conducted on top of it validate the proposed models and theoretical results.

10. Acknowledge

This research was supported in part by NSF grants CNS 2128378, CNS 2107014, CNS 1824440, CNS 1828363, CNS 1757533, CNS 1629746, and CNS 1651947.

References

[1] Rosen JB. Existence and uniqueness of equilibrium points for concave n-person games. Econometrica: Journal of the Econometric Society. 1965;:520–534.
[2] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016. p. 770–778.
[3] Huang G, Liu Z, Van Der Maaten L, et al. Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2017. p. 4700–4708.
[4] Ziad A. Nash equilibria in pure strategies. Bulletin of Economic Research. 2003;55(3):311–317.
[5] Wang X, Zha X, Ni W, et al. Survey on blockchain for internet of things. Computer Communications. 2019;136:10–29.
[6] Christidis K, Devetsikiotis M. Blockchains and smart contracts for the internet of things. IEEE Access. 2016;4:2292–2303.
[7] Cita ; ???? Available from: https://github.com/cryptape/cita.

[8] Recht B, Re C, Wright S, et al. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. Advances in Neural Information Processing Systems. 2011;24:693–701.
[9] Li T, Sahu AK, Zaheer M, et al. Federated optimization in heterogeneous networks. arXiv preprint arXiv:181206127. 2018;.
[10] Charles Z, Papailiopoulos D. Gradient coding using the stochastic block model. In: 2018 IEEE International Symposium on Information Theory (ISIT); IEEE; 2018. p. 1998–2002.
[11] Corinzia L, Buhmann JM. Variational federated multi-task learning. arXiv preprint arXiv:190606268. 2019;.
[12] Eichner H, Koren T, McMahan HB, et al. Semi-cyclic stochastic gradient descent. arXiv preprint arXiv:190410120. 2019;.
[13] Khodak M, Balcan MFF, Talwalkar AS. Adaptive gradient-based meta-learning methods. In: Advances in Neural Information Processing Systems; 2019. p. 5917–5928.
[14] Li T, Sahu AK, Talwalkar A, et al. Federated learning: Challenges, methods, and future directions. IEEE Signal Processing Magazine. 2020;37(3):50–60.
[15] Bonawitz K, Eichner H, Grieskamp W, et al. Towards federated learning at scale: System design. arXiv preprint arXiv:190201046. 2019;.
[16] Stich SU. Local sgd converges fast and communicates little. arXiv preprint arXiv:180509767. 2018;.
[17] Zhang S, Choromanska AE, LeCun Y. Deep learning with elastic averaging sgd. Advances in Neural Information Processing Systems. 2015;28:685–693.
[18] Kang J, Xiong Z, Niyato D, et al. Reliable federated learning for mobile networks. IEEE Wireless Communications. 2020;27(2):72–80.
[19] Kim H, Park J, Bennis M, et al. On-device federated learning via blockchain and its latency analysis. arXiv preprint arXiv:180803949. 2018;.
[20] Zhao Y, Zhao J, Jiang L, et al. Mobile edge computing, blockchain and reputation-based crowdsourcing iot federated learning: A secure, decentralized and privacy-preserving system. arXiv preprint arXiv:190610893. 2019;.
[21] Singer Y, Mittal M. Pricing mechanisms for crowdsourcing markets. In: Proceedings of the 22nd International Conference on World Wide Web; 2013. p. 1157–1166.
[22] Singla A, Krause A. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In: Proceedings of the 22nd International Conference on World Wide Web; 2013. p. 1167–1178.
[23] Goel G, Nikzad A, Singla A. Mechanism design for crowdsourcing markets with heterogeneous tasks. In: HCOMP; 2014.
[24] Zeng R, Zhang S, Wang J, et al. Fmore: An incentive scheme of multi-dimensional auction for federated learning in mec. arXiv preprint arXiv:200209699. 2020;.
[25] Kang J, Xiong Z, Niyato D, et al. Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory. IEEE Internet of Things Journal. 2019;6(6):10700–10714.
[26] Fishburn PC. Utility theory. Management Science. 1968;14(5):335–378.
[27] Wu J, Lu M, Li F. Utility-based opportunistic routing in multi-hop wireless networks. In: IEEE 28th International Conference on Distributed Computing Systems; IEEE; 2008. p. 470–477.
[28] Cunial F. Price of anarchy ; ???? Available from: https://web.archive.org/web/20080910060646/http://wiki.cc.gatech.edu/theory/index.php/Price$_o f_a narchy$.
[29] Fiat A, Kaplan H, Levy M, et al. Strong price of anarchy for machine load balancing. In: International Colloquium on Automata, Languages, and Programming; Springer; 2007. p. 583–594.
[30] Johari R. The price of anarchy and the design of scalable resource allocation mechanisms. Algorithmic Game Theory. 2007;:543–568.
[31] Youn H, Gastner MT, Jeong H. Price of anarchy in transportation networks: efficiency and optimality control. Physical Review Letters. 2008;101(12):128701.
[32] Ye D, Chen J. Non-cooperative games on multidimensional resource allocation. Future Generation Computer Systems. 2013;29(6):1345–1352.

[33] Ye D, Chen L, Zhang G. On the price of anarchy of two-stage machine scheduling games. Journal of Combinatorial Optimization. 2019;:1–20.
[34] Chen YJ, Zhang J. Design of price mechanisms for network resource allocation via price of anarchy. Mathematical Programming. 2012;131(1-2):333–364.
[35] Marden JR, Philips M. Optimizing the price of anarchy in concave cost sharing games. In: 2017 American Control Conference (ACC); IEEE; 2017. p. 5237–5242.
[36] Peng Z, Xu J, Chu X, et al. Vfchain: Enabling verifiable and auditable federated learning via blockchain systems. IEEE Transactions on Network Science and Engineering. 2021;.
[37] Lu Y, Huang X, Zhang K, et al. Blockchain empowered asynchronous federated learning for secure data sharing in internet of vehicles. IEEE Transactions on Vehicular Technology. 2020;69(4):4298–4311.
[38] Lu Y, Huang X, Dai Y, et al. Blockchain and federated learning for privacy-preserved data sharing in industrial iot. IEEE Transactions on Industrial Informatics. 2019;16(6):4177–4186.
[39] Desai HB, Ozdayi MS, Kantarcioglu M. Blockfla: Accountable federated learning via hybrid blockchain architecture. In: Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy; 2021. p. 101–112.