An Incentive Auction for Heterogeneous Client Selection in Federated Learning

Jinlong Pang, Jieling Yu, Ruiting Zhou, *Member, IEEE*, and John C.S. Lui, *Fellow, IEEE*

Abstract—Federated Learning (FL) is a new distributed machine learning (ML) approach which enables thousands of mobile devices to collaboratively train artificial intelligence (AI) models using local data without compromising user privacy. Although FL represents a promising computing paradigm, such training process can not be fully realized without an appropriate economic mechanism that incentivizes the participation of heterogeneous clients. This work targets social cost minimization, and studies the incentive mechanism design in FL through a procurement auction. Different from existing literature, we consider a practical scenario of FL where clients are selected and scheduled at different global iterations to guarantee the completion of the FL job, and capture the distinct feature of FL that the number of global iterations is determined by the local accuracy of all participants to balance between computation and communication. Our auction framework A_{FL} first decomposes the social cost minimization problem into a series of winner determination problems (WDPs) based on the number of global iterations. To solve each WDP, A_{FL} invokes a greedy algorithm to determine the winners, and a payment algorithm for computing remuneration to winners. Finally, A_{FL} returns the best solution among all WDPs. We carried out theoretical analysis to prove that A_{FL} is truthful, individual rational, computationally efficient, and achieves a near-optimal social cost. We further extend our model to consider multiple FL jobs with corresponding budgets and propose another efficient algorithm A_{FL-M} to solve the extended problem. We conduct large-scale simulations based on the real-world data and testbed experiments by adopting FL frameworks FAVOR and CoCoA. Simulation and experiment results show that both A_{FL} and A_{FL-M} can reduce the social cost by up to 55% compared with state-of-the-art algorithms.

Index Terms—Federated learning, incentive mechanism, auction

1 Introduction

The emergence of federated learning (FL) provides a new computing paradigm for artificial intelligence (AI) and its application. Traditional machine learning (ML) trains AI models centrally, which is privacy-intrusive, especially for mobile devices which contain owners' privacy-sensitive data [1], [2]. Compared to the centralized training process, FL is a decentralized training approach which distributes ML jobs to thousands of geo-distributed mobile devices (*a.k.a.* clients) [3], [4]. Mobile devices act as the computing nodes to collaboratively train a ML model using local data, without the risk of privacy disclosure. Major enterprises have launched FL projects. For example, Gboard, the Google Keyboard on Android, is adopting the FL process to make typing faster and easier. Mobile phones locally store users' typing preference every time when Gboard shows a suggested query. FL trains Gboard's query suggestion model using history on device to improve user experience in the next iteration [5].

To fully realize the potential of FL in practice, two types of challenges need to be addressed: *technical* and *economical*. *First*, on the technical side, both computation and communication are the core challenges. The learning process in FL relies on frequent communication between the cloud server and mobile clients to update model, until the model converges [4]. To achieve a lower local accuracy[1], mobile clients spend more computation time to train their local models. Given a required global accuracy of the model, the number of communication rounds is proportional to the local accuracy achieved by all clients [7], [8]. Thus, how to balance between computation and communication time through the selection of clients with their local accuracy while guaranteeing fast convergence of the model becomes a vital problem. *Second*, on the economic side, incentive mechanism design is an indispensible enabling technology for FL. The training process in FL is iterative, and needs thousands of clients to work collaboratively and continuously [4], [9]. However, it is not always practical to assume that mobile clients are voluntary to fully participate in the complete training process, since mobile clients consume their own

- *Jinlong Pang and Jieling Yu are with the Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, Hubei 430072, China. E-mail: {jinlongpang, yjling}@whu.edu.cn.*
- *Ruiting Zhou is with the Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, Hubei 430072, China, and also with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong. E-mail: ruitingzhou@whu.edu.cn.*
- *John C.S. Lui is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong. E-mail: cslui@cse.cuhk.edu.hk.*

Manuscript received 23 June 2021; revised 29 March 2022; accepted 7 June 2022. Date of publication 14 June 2022; date of current version 31 August 2023. This work was supported in part by the NSFC under Grants 62072344 and U20A20177. The work of John C.S. Lui was supported in part by the RGC under Grant GRF-14200321.
(Corresponding author: Ruiting Zhou.)
Digital Object Identifier no. 10.1109/TMC.2022.3182876

1. Here, local accuracy θ and global accuracy ε represent the relative gradient difference of loss function between two iterations [6], *i.e.*, $||\nabla F(w^{(t)})|| \leq \theta||\nabla F(w^{(t-1)})||$ and $||\nabla J(w^{(t)})|| \leq \varepsilon||\nabla J(w^{(t-1)})||$, where $F(w)$ and $J(w)$ are the loss function of local and global model, respectively.

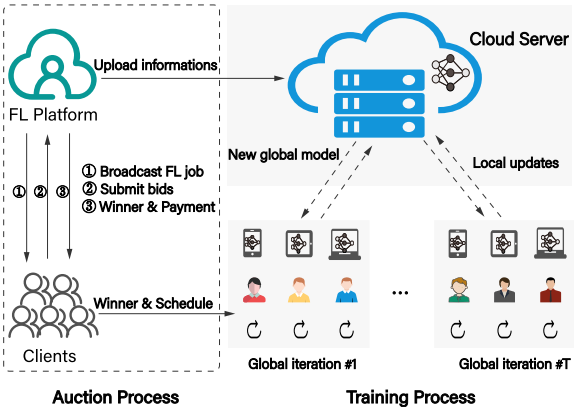

Fig. 1. An illustration of federated learning auction.

resources such as battery and GPU to calculate local model updates. Moreover, clients have their own schedule and may only participate in some particular time periods. Therefore, incentive mechanisms which pay rewards to compensate the cost of clients are the essential financial catalyst for making FL a reality.

To overcome aforementioned challenges, one needs to capture the distinct feature of FL while designing incentive mechanisms, *i.e.*, the relation between global accuracy and local accuracy among many mobile clients. Most existing research in FL focuses on the technical side, and investigates how to improve the training efficiency or reduce energy cost [6], [10], [11], [12]. There are only a few studies on the incentive mechanism design in FL. Most work in FL assume that the same set of clients can fully participate in the whole FL training process from beginning to end, and select energy-efficient clients to achieve fairness or utility maximization [13], [14], [15], [16], which we will discuss in details in Section 2. In fact, incentive mechanism design has been well-studied in other related fields (*e.g.*, mobile crowdsensing or crowdsourcing) by leveraging several approaches mentioned in Section 2. On behalf of one of the representative incentivizing approaches, auction already showed its superiority. In contrast to other incentive mechanism approaches (*e.g.*, contract theory [17], [18]) in which clients can only decide whether or not to accept the contracts, auction enables clients to bid for any combinations of resources. Meanwhile, traditional fixed pricing cannot exactly capture the flexible supply-demand relationship between clients and the cloud server due to the heterogeneous capacities of devices. Consequently, overpricing and underpricing routinely occur, jeopardizing the cloud server's profit as well as the system's utility. In contrast, auction is a natural approach to balance supply and demand, and automatically discover the right price, so that the cloud server can select clients with the lowest cost. In addition, auction-based framework can simultaneously guarantee individual rationality and truthfulness. Therefore, in this work, we propose a solution for incentivizing participation in FL as a procurement auction, A_{FL}. As shown in Fig. 1, the procurement auction consists of multiple sellers (mobile clients) and a single buyer (the cloud server). The goal of the auction design is social cost minimization while guaranteeing computational efficiency, truthfulness and individual rationality. Different from existing literature, we describe a richer and practical model of FL. We select and schedule clients at different global iterations to guarantee the completion of the FL job, and determine the number of global iterations (communication rounds) by the local accuracy of all participants to balance between computation and communication. We summarize our main contributions as follows.

First , we model and formulate the social cost minimization problem in FL as an integer linear program (ILP), and prove it is NP-hard. Different from existing literature that only determines winners (or selected mobile clients), we also need to decide how to schedule the participation of winners and the number of global iterations. To address the challenge introduced by the variation on the number of global iterations, our A_{FL} first calculates a range for the number of global iterations. Then for each fixed number of global iterations within the range, we formulate a winner determination problem (WDP). This way, A_{FL} decomposes the original optimization problem into a series of WDPs. To reduce computation, we exclude those bids which violate the communication round and computation time constraints, and further form a qualified bids set for each WDP. A_{FL} next invokes an algorithm A_{winner} to solve each WDP and finally announces the auction results, including the winning bids which generate the minimum social cost and the payment to winners.

Second , to determine and schedule winners for each WDP, we first reformulate each WDP to a new ILP by using compact exponential technique [19], which is a packing-type ILP with an exponential number of variables corresponding to valid schedules. This exponential-sized ILP and its dual are the foundation of algorithm design and analysis. We show that the new ILP can be solved by a greedy algorithm A_{winner}. A_{winner} iteratively selects a client with a schedule which can cover available global iterations at the lowest average cost, until there are enough participants in the winner set. Furthermore, a payment scheme based on the critical value rule [20], [21] is proposed as a subroutine of A_{winner} to ensure truthfulness and individual rationality. We conduct rigorous theoretical analysis to show that A_{FL} is truthful, individual rational and computationally efficient. Furthermore, we adopt the primal-dual theory to prove that A_{FL} achieves a good approximation ratio in social cost.

Third , we extend to discuss one new realistic scenario where there are multiple FL jobs with budget functions. To illustrate the preference on the FL job's completion time, the budget deceases with the completion time monotonically. In this scenario, there are extreme cases where some clients' bids are selected by different FL jobs at the same time. To address it, from the prospective of clients, we consider rescheduling clients for FL jobs via a greedy fashion. Furthermore, to reduce the time complexity, we apply binary search to narrow the range of the number of global iterations without compromising social cost greatly. The new auction is presented in algorithm A_{FL-M} to solve the extended problem.

Last but not the least , we evaluate the performance of A_{FL} through large-scale simulations based on real-world data. Numerical results demonstrate that A_{FL} always outperforms three benchmark algorithms. Moreover, A_{FL} produces a close-to-optimal social cost with a small ratio (< 1.3), and reduces the social cost by 65%, 85%, 300%, compared with A_{online} [19], *GAA* [22] and *FedAvg* [4], respectively. We further conduct the simulations to examine

A_{FL-M}. A_{FL-M} can achieve a near performance compared with A_{FL}. Then we conduct extensive testbed experiments by adopting FL frameworks FAVOR [23] and CoCoA [8] to evaluate the performance of A_{FL-M}. The results of experiments show that A_{FL-M} reduces the social cost by at least 55% compared with three benchmark algorithms. Moreover, we further discuss the accuracy of the FL job under different levels of non-IID data/parallelism.

In the rest of the paper, related work is given in Section 2. The preliminary and system model of FL are introduced in Sections 3 and 4, respectively. The procurement auction is presented and analyzed in Sections 5 and 6. Section 7 considers a more practical multiple jobs scenario where each FL job has a budget function. The results of simulations and experiments are shown in Sections 8 and 9. Section 10 concludes the paper.

2 Related Work

Federated Learning. In FL, majority of researchers focus on learning algorithm design with verifiable convergence analysis, but ignore practical challenges and economic incentives. Several papers studied FL based on practical scenarios. Smith *et al.* [10] consider practical systems challenges in FL such as straggler effort and random drops, and propose an efficient optimization method to address these issues. Tran *et al.* [6] focus on the trade-off between communication and computation cost, and obtain the optimal number of communication rounds, accuracy-level and minimum energy cost. Nishio *et al.* [11] present a FL protocol, which selects as many clients as possible to maximize training efficiency under the stragglers' effort caused by heterogeneous resources. Considering devices' computation capacity and limited networking resources, Wang *et al.* [12] design a control algorithm to dynamically adapt aggregation frequency. To speed up the convergence of FL, Wang *et al.* [23] select clients (mobile devices) with non-IID data through deep reinforcement learning (DRL), rather than selecting randomly like FedAvg [4]. To solve the problem of network heterogeneity and local data overlap between devices in FL, Wang *et al.* [24] develop an optimization methodology that uses intelligent device sampling. From the perspective of energy efficiency, Zhan *et al.* [25] design an experience-driven method based on DRL to control devices' CPU-cycle frequency in a synchronized setting. Luo *et al.* [26] design an adaptive FL to minimize costs (time and energy costs) while ensuring convergence. Deng *et al.* [27] focus on users' incentive and quality of model aggregation, and propose a novel federated learning system with quality awareness: FAIR. Wang *et al.* [28] pay attention to the problem of long training time and limited resources in FL, and propose a hierarchical aggregation, resource-efficient federated learning. For the accuracy reduction and model instability caused by poor samples, Li *et al.* [29] present an efficient and privacy-preserving high-quality sample selection system. Aiming at the security risks of the malicious attacks and reverse analysis in FL, Wei *et al.* [30] propose a user-level differential privacy algorithm to effectively improve FL's privacy protection level. For the above work, their aim is to improve the performance, privacy guarantee or achieve cost minimization, but neglect the problem of how to incentivize the participation of clients.

Incentive Mechanisms. In mobile crowdsourcing or crowdsensing system, there has been a long study on incentive mechanisms, especially using contract theory [17], [18], auction [31], [32] and game theory [33], [34]. However, there are only a few studies on the incentive mechanism design for implementing FL. Kang *et al.* [15] aim to address unreliable updates, and propose a contract-theoretic method to motivate clients that have a high reputation and high-qualified data to do the update. Ding *et al.* [35] analyze the incentive mechanism design for FL by using contract theory when considering multi-dimensional privacy information. Pandey *et al.* [14] develop a Stackelberg game-based framework to incentivize clients to participate in training to achieve utility maximization. Toyoda *et al.* [13] design an incentive-aware mechanism for a blockchain-enabled FL platform by using contest theory to guarantee fairness. Zeng *et al.* [16] consider multi-dimensional resources and present an incentive framework based on game theory to achieve utility maximization. Weng *et al.* [36] customize a privacy-protecting, truthful, accuracy-boosting incentive mechanism based on Bayesian game theory. Le *et al.* [37] develop an auction-based incentive mechanism to stimulate clients in the wireless network scenario. Based on the goal of filtering untrustworthy or low-quality learning parameters from malicious or inactive learners, Lin *et al.* [38] propose a federated edge learning incentive mechanism based on social networks. Tang *et al.* [39] formulate an incentive mechanism for cross-silo federation learning to maximize social welfare. Jiao *et al.* [40] maximize the social welfare of a typical FL service system and formulate two incentive mechanisms. Lim *et al.* [41] present a resource allocation and incentive mechanism design framework for hierarchical federated learning (HFL) by jointly leveraging evolutionary game theory and deep learning-based auction. Similarly, Ng *et al.* [42] propose a hierarchical two-level incentive mechanism design to allocate the resources of the data owners and FL workers so that completing the coded federated learning (CFL) tasks. Above studies are all based on an impractical assumption that the same set of clients can fully participate in the complete process from beginning to end. In addition, they fixed the number of global iterations at the beginning. Different from the above literature, we select cost-efficient clients and schedule them at different global iterations. In addition, to balance between computation and communication, we also determine that the number of global iterations is affected by winners' local accuracy.

3 Preliminary of Federated Learning

The learning process in FL [8] relies on the iterative interaction between the cloud server and clients. In each global iteration: i) each selected client trains its local model on its local dataset for a number of local iterations to achieve a desirable local accuracy; ii) then each client returns its local model update to the server; iii) the server aggregates all local model updates and sends back the global model update to clients. The above process terminates until the global model accuracy reaches a predefined threshold. For strongly convex optimization problem in federated learning (*i.e.*, loss function), the upper bound on the number of global iterations (T_g) can be expressed according to the

TABLE 1
List of Notations

I	# of clients	$\mathcal{X}$	integer set $\{1,\ldots,X\}$
p_i	payment to client i	ε	global accuracy
J	# of submitted bids	$\mathcal{S}$	winner set
$V(\varepsilon)$	actual reward the cloud server received		
T	maximum number of global iterations		
T_g	# of global iterations		
B_{ij}	bid information of client i's j-th bid		
b_{ij}	asking price of client i's j-th bid		
v_{ij}	true cost of client i's j-th bid		
θ_{ij}	local accuracy of client i's j-th bid		
a_{ij}	starting global iteration of the available time period of client i's j-th bid		
d_{ij}	ending global iteration of the available time period of client i's j-th bid		
c_{ij}	# of participation rounds of client i's j-th bid		
x_{ij}	whether or not to accept client i's j-th bid		
$y_i(t)$	whether or not to schedule client i at t-th iteration		
t_i^{cmp}	computation time required for client i to perform one local iteration		
t_i^{com}	communication time required for client i in one global iteration		
$t_{\max}$	duration of a single global iteration		
$T_l(\theta_{ij})$	# of local iterations with local accuracy θ_{ij}		
$\mathcal{L}_i$	the set of feasible schedules of client i		
ρ_{il}	client i's bidding price with schedule l		
z_{il}	whether or not to accept client i's schedule l		
$H_{\hat{T}_g}$	a harmonic number, which is equal to $\sum_{t=1}^{\hat{T}_g}\frac{1}{t}$		
ω	an auxiliary variable defined in line 18 of Alg. 2, equals $\max_{t\in\hat{\mathcal{T}}_g}\omega_t$		

definitions in [8], [10], as follows:

$$T_g = \frac{\mathcal{O}(\log(\frac{1}{\varepsilon}))}{1-\theta_{\max}}, \tag{1}$$

where $\varepsilon \in [0,1]$ is the predefined global accuracy and $\theta_{\max} \in [0,1]$ is the maximum local accuracy among all clients. Let $T_l(\theta_i)$ denote the number of local iterations for client i to achieve its local accuracy θ_i, which can be defined as follows [8]:

$$T_l(\theta_i) = \eta \cdot \log\left(\frac{1}{\theta_i}\right), \tag{2}$$

where η is a positive constant. For ease of exposition, our work focuses more on the system challenges in FL, (*i.e.*, fault-tolerance or stragglers efforts) rather than the data characteristics of FL (*i.e.*, non-IID). So we basically assume that all clients' data follow an identical and independent distribution. Therefore, it is reasonable to believe that one "dropped" client can be replaced by another client due to the IID data characteristic. In this regard, our work actually uses a simplified version of the result of [10]. That is, we do not need to use multi-task learning to handle the statistical challenge (mentioned in [10]), so all clients' training models are the same.

4 System Model

4.1 System Overview

As shown in Fig. 1, we consider a typical scenario of FL which involves a cloud server and a set of clients (*e.g.*, smartphone or personal computer). On a FL platform, the cloud server first broadcasts the information of a FL job to all clients, including the maximum number of global iterations T (*i.e.*, communication rounds), the duration of one global iteration $t_{\max}$[2], the number of participating clients in each global iteration K[3] and the expected global accuracy ε. Let $V(\varepsilon)$ denote the actual reward the cloud server can receive with global accuracy ε. To incentivize clients, a "*procurement auction*" is applied where the server acts as the auctioneer and each client submits a bid for job participation. After collecting all bids, the server determines and pays the selected winners, and then schedules them to collaboratively execute the FL job. Let $\mathcal{X}$ denote the integer set $\{1,2,\ldots,X\}$. Important notations are listed in Table 1.

Once selected, each winner will participate in by submitting its partial update in specific global iteration according to the schedule, in order to achieve the pre-defined global accuracy ε. During the process of training, one global iteration is a duration when clients use their private personal data to train local models repeatedly until achieving a local model accuracy θ.

4.2 Auction Model

Bidding Information . Let I denote the number of available clients. In practice, a client may not be able to fully participate in the entire training process due to many factors, *e.g.*,

2. To avoid the straggler effort caused by the heterogeneous capacities of clients, we apply a simple synchronous scheme [43]. The FL platform limits the maximum duration of a single global iteration to $t_{\max}$, and calculates the number of global iterations T based on the expected maximum completion time of the FL job and $t_{\max}$.

3. Typically, the value of K is fixed and ranges from 10 to 200 [4], [8], [10], [23].

battery level or personal schedule [43]. Furthermore, a client values different periods and local accuracy differently. Therefore, we assume that each client i submits up to J bids, and client i's j-th bid (B_{ij}) is expressed as a tuple:

$$B_{ij} = \{b_{ij}, \theta_{ij}, [a_{ij}, d_{ij}], c_{ij}\}_{\forall j \in \mathcal{J}}, \tag{3}$$

where b_{ij} is the "claimed" cost that client i wants to charge for the service. θ_{ij} is the local accuracy. $[a_{ij}, d_{ij}]$ is the available time period within $\mathcal{T}$, which starts and ends at a_{ij}-th and d_{ij}-th global iteration. In period $[a_{ij}, d_{ij}]$, client i can only participate c_{ij} number of global iterations, which is limited by its battery level, and calculated based on θ_{ij}. Let v_{ij} be the "true" cost of client i's j-th bid.

Decision Variables . After receiving all bids from the clients, the cloud server will make the following decisions: i) $T_g \in \{1, 2, \ldots, T\}$, the number of global iterations, ii) $x_{ij} \in \{0, 1\}$, whether or not to accept client i's j-th bid, and if so, iii) $y_i(t) \in \{0, 1\}$, whether or not to schedule client i at t-th global iteration, and iv) p_i, payment to client i.

Constraints and Assumptions . *(1) Requirement of FL training:* The training process of federated learning [8] requires a fixed number of clients. The above requirement can be modeled as constraint (4), which ensures that at least K clients are selected for each global iteration.

$$\sum_{i \in \mathcal{I}} y_i(t) \geq K, \quad \forall t \in \mathcal{T}_g. \tag{4}$$

(2) Accuracy Requirement: To guarantee the theoretical accuracy requirement mentioned in Section 3, we rewrite the Eq. (1) as one specific constraint, shown as (5). For the ease of presentation, $\mathcal{O}(\log(\frac{1}{\varepsilon}))$ is normalized to 1 when we consider a fixed global accuracy ε. So constraint (5) calculates the number of global iterations T_g, according to the maximum local accuracy $\theta_{\max}$ among the winners.

$$T_g \geq \frac{1}{1 - \theta_{ij} x_{ij}}, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}. \tag{5}$$

(3) Bid Restriction: the number of participation rounds for each client (*i.e.*, $\sum_{t \in \mathcal{T}_g} y_i(t)$) is equal to the number of claimed rounds in its bid, as shown in constraint (6).

$$\sum_{t \in \mathcal{T}_g} y_i(t) = \sum_{j \in \mathcal{J}} c_{ij} x_{ij}, \quad \forall i \in \mathcal{I}. \tag{6}$$

(4) The Duration of Global Iteration: The time for client i's j-th bid to compute the local update in one global iteration consists of two parts: computation time $T_l(\theta_{ij}) t_i^{cmp}$ and communication time t_i^{com}. For simplicity, suppose that when a client registered at the FL platform, the cloud server can access its information. So t_i^{cmp} and t_i^{com} can be considered as constants. This time duration constraint for one global iteration can be formulated as:

$$x_{ij} \cdot (T_l(\theta_{ij}) t_i^{cmp} + t_i^{com}) \leq t_{\max}, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}. \tag{7}$$

(5) Decision Variable Restriction: The relationship between $y_i(t)$ and x_{ij} is formulated by constraint (8). That is, one client can participate in training only when it is selected by the cloud server ($x_{ij=1}$).

$$y_i(t) = 1 \text{ only if } x_{ij} = 1, \quad t \in [a_{ij}, d_{ij}], \forall i \in \mathcal{I}, \forall j \in \mathcal{J}. \tag{8}$$

(6) XOR Bidding Rule: Even each client submits up to J bids, only one bid can be accepted. This is because each client can only participate in one time period due to its battery capacity. Therefore, constraint (9) specifies that at most one bid of each client can be accepted.

$$\sum_{j \in \mathcal{J}} x_{ij} \leq 1, \quad \forall i \in \mathcal{I}. \tag{9}$$

Auction Preliminary . We next introduce some definitions in auction design. The cloud server's utility is:

$$u_{server} = V(\varepsilon) - \sum_{i \in \mathcal{I}} p_i, \tag{10}$$

Then client i's utility is:

$$u_i = \begin{cases} p_i - v_{ij}, & \text{if } x_{ij} = 1 \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

In general, clients are selfish and tend to maximize their own utilities. They may even lie about their true cost to get a higher utility. We instead focus on the utilities of the entire FL system, and target *social welfare maximization*. Therefore, it is necessary to elicit truthful bids from clients.

Definition 1. *(Truthful in bidding price): An auction is truthful in bidding price if and only if each client's utility is maximized when it bids with its true cost,* i.e., *for all* $b_{ij} \neq v_{ij}$, $u_i(v_{ij}) \geq u_i(b_{ij})$.

Definition 2. *(Individual Rationality): An auction is individual rational if each client's utility is non-negative,* i.e., $u_i(b_{ij}) \geq 0$.

Definition 3. *(Social Welfare, Social cost): The social welfare of the FL system is the aggregate utility of the cloud sever and clients, and equals* $V(\varepsilon) - \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} v_{ij} x_{ij}$. *When* $V(\varepsilon)$ *is a fixed value, one can ignore it. Note that in the optimizing process, maximizing social welfare is equivalent to minimizing the social cost,* i.e., $\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} v_{ij} x_{ij}$.

4.3 Social Cost Minimization Problem

Problem Formulation . Under truthful bidding ($b_{ij} = v_{ij}$), the social cost minimization problem can be formulated into the following integer linear program (ILP):

$$\text{minimize} \qquad \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} b_{ij} x_{ij} \tag{12}$$

subject to:

$$(4) - (9),$$

$$x_{ij}, y_i(t) \in \{0, 1\}, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall t \in \mathcal{T}_g, \tag{12g}$$

$$T_g \in \{1, 2, \ldots, T\}. \tag{12h}$$

Challenges. Note that even a simplified version of ILP (12) without constraints (5), (7) and (12 h) is still NP-hard, which is equivalent to the set cover problem [44]. The challenge becomes more complicated when this problem involves a non-trivial variable T_g which relates to all winners. Moreover, two sets of binary variables determine clients' participation schedules, and eventually affect the total social cost.

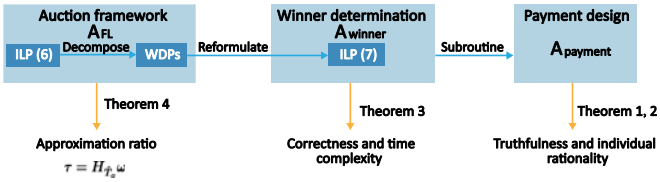

Fig. 2. Main idea of FL auction A_{FL}.

5 Auction Design

5.1 Overview of Auction Design

Algorithmic Idea. To solve the ILP (12), we present an auction framework, A_{FL}, to determine the winning bids and corresponding schedules to minimize the social cost with a bounded approximation ratio. The high level algorithmic idea of A_{FL} is shown in Fig. 2.

i. A_{FL} first computes the range of T_g according to clients' local accuracy. Then for each fixed $\hat{T}_g$ within the range, it formulates a winner determination problem (WDP). The input of each WDP is a qualified bids set which satisfies constraints (5) and (7). A_{FL} then decomposes ILP (12) into several WDPs. A_{FL} next calls A_{winner} to solve each WDP and finally outputs the winning bids which generate the minimum social cost.

ii. In Section 5.2, we show how to determine the winners for each WDP. We first encode x_{ij} and $y_i(t)$ into one variable and reformulate each WDP into ILP (13). To solve it, we design an approximation algorithm A_{winner} based on a greedy strategy to select winning bids and schedule clients' participation.

iii. In Section 5.3, we show how to charge winners for each WDP. We propose a payment algorithm $A_{payment}$ which is a subroutine of A_{winner} based on the critical value rule [20], [21].

Algorithm 1. FL Auction Framework A_{FL}

Input: T, K, $B_{ij}, \forall i, j$;
Output: T_g^*, $minicost$, $\mathcal{S}^*$;
1: Initialize $t_{ij} = T_l(\theta_{ij})t_i^{cmp} + t_i^{com}, \forall i, j$; $\mathcal{S}^* = \mathcal{P}^* = \mathcal{J}_{\hat{T}_g} = \varnothing$, $minicost = \infty$;
2: Find the minimum local accuracy $\theta_{\min}$ of all bids;
3: $T_0 = \lfloor 1/(1-\theta_{\min}) \rfloor$;
4: **for** $\hat{T}_g = T_0$ to T **do**
5: $\theta_{\max} = \lceil 1 - 1/\hat{T}_g \rceil$;
6: $\mathcal{J}_{\hat{T}_g} = \{(i,j)_{\forall i,j} | \theta_{ij} \le \theta_{\max} \& t_{ij} \le t_{\max} \& a_{ij} + c_{ij} \le \hat{T}_g\}$;
7: $(\mathcal{S}, \mathcal{P}, cost(\hat{T}_g)) = A_{winner}(\mathcal{J}_{\hat{T}_g}, \hat{T}_g, K)$;
8: **if** $cost(\hat{T}_g) < minicost$ **then**
9: $T_g^* = \hat{T}_g$, $minicost = cost(\hat{T}_g)$, $\mathcal{S}^* = \mathcal{S}$, $\mathcal{P}^* = \mathcal{P}$;
10: **end if**
11: **end for**
12: **for all** $x_{ij} == 1, \forall x_{ij} \in \mathcal{S}^*$ **do**
13: Accept client i's j-th bid and schedule client i according to $y_i(t) \in l_{ij}$; Pay $p_i \in \mathcal{P}^*$ to client i;
14: **end for**

Auction Framework. Our FL auction A_{FL} is presented in Algorithm 1. Let t_{ij} be the time for client i's j-th bid to compute and transmit the local update in one global iteration. Line 1 initializes all variables. Given the local accuracy of all bids, A_{FL} selects the minimum local accuracy to compute the initial value T_0 for T_g in lines 2-3. Then, A_{FL} enumerates the number of global iterations $\hat{T}_g$ from T_0 to T and computes the feasible maximum local accuracy $\theta_{\max}$ for different $\hat{T}_g$ according to Eq. (1) in lines 4-5. Next, $\theta_{\max}$, $t_{\max}$ and $\hat{T}_g$ are used to get a set of qualified bids $\mathcal{J}_{\hat{T}_g}$ for A_{winner} (line 6). In line 7, algorithm A_{winner} returns winners' set $\mathcal{S}$, the payment set $\mathcal{P}$ and the corresponding social cost $cost(\hat{T}_g)$. A_{FL} then compares the resulting costs at different $\hat{T}_g$, and records the best solution which achieves the minimum social cost (lines 8-10). Finally, A_{FL} announces the auction result in lines 12-14.

5.2 Winner Determination

To solve each WDP and determine the winners, we next present the algorithmic design of A_{winner}.

1) Problem Reformulation

For each fixed $\hat{T}_g$, there is a WDP with a qualified bid set $\mathcal{J}_{\hat{T}_g}$. Recall that each WDP is equivalent to a simplified version of ILP (12) without constraints (5), (7) and (12 h). Note that two decision variables x_{ij} and $y_i(t)$ have a natural precedence correlation. To address this problem, we apply compact exponential technique [19] to reformulate the WDP to the following ILP (13) by encoding x_{ij} and $y_i(t)$ into a new decision variable z_{il}.

$$\text{minimize} \quad \sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}_i} \rho_{il} z_{il} \tag{13}$$

subject to:

$$\sum_{i \in \mathcal{I}} \sum_{l: y_i(t) \in l} z_{il} \ge K, \quad \forall t \in \hat{\mathcal{T}}_g, \tag{13a}$$

$$\sum_{l \in \mathcal{L}_i} z_{il} \le 1, \quad \forall i \in \mathcal{I}, \tag{13b}$$

$$z_{il} \in \{0,1\}, \quad \forall l \in \mathcal{L}_i, \forall i \in \mathcal{I}. \tag{13c}$$

In the above ILP (13), $\mathcal{L}_i$ is the schedule set of client i. A feasible schedule l is a vector $l = \{\{x_{ij}\}_{\forall (i,j) \in \mathcal{J}_{\hat{T}_g}}, \{y_i(t)\}_{\forall i,t}\}$ which satisfies constraints (6) and (8). The value of ρ_{il} equals the corresponding b_{ij} based on l. z_{il} denotes whether or not to select client i's schedule l. Note that the number of feasible schedules z_{il} for client i is exponential, due to combinatorial property of variables x_{ij} and $y_i(t)$ (*i.e.*, the number of feasible schedules for client i is up to $\sum_j \binom{d_{ij}-a_{ij}}{c_{ij}}$). Constraint (13 a) is equivalent to constraint (4). And constraint (13 b) ensures that one client can be selected according to at most one schedule. It is clear that a feasible solution to ILP (13) is also a feasible solution to the WDP, and vice versa, with the same objective value.

Dual Problem. In order to analyze the performance of A_{winner}, we formulate the dual problem of ILP (13) by relaxing the integer constraint (13 c) into $0 \le z_{il} \le 1$, and introduce dual variables $g(t)$, q_i and λ_{il} to constraint (13 a), (13 b) and $z_{il} \le 1$, respectively. Then, the dual problem of relaxed ILP (13) is:

$$\text{maximize} \quad \sum_{t \in \hat{\mathcal{T}}_g} Kg(t) - \sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}_i} \lambda_{il} - \sum_{i \in \mathcal{I}} q_i \tag{14}$$

subject to:

$$\sum_{t:y_i(t)\in l} g(t) - \lambda_{il} - q_i \le \rho_{il}, \quad \forall l \in \mathcal{L}_i, \forall i \in \mathcal{I}, \tag{14a}$$

$$g(t), \lambda_{il}, q_i \ge 0, \quad \forall t \in \hat{\mathcal{T}}_g, \forall l \in \mathcal{L}_i, \forall i \in \mathcal{I}. \tag{14b}$$

2) Winner Determination and Scheduling

Main Idea. To get a feasible solution of the exponential-sized ILP (13), we design an efficient algorithm A_{winner} which selects schedules iteratively based on a greedy strategy. We say the t-th global iteration is *available* if the number of selected clients in the t-th global iteration is less than K. A_{winner} starts with an empty set. In each iteration, A_{winner} selects a client with a schedule which can cover *available* global iterations at the lowest *average cost*. Then A_{winner} adds the selected client with its corresponding schedule to the winner set. This process terminates until there are enough participants in the winner set.

Algorithm 2. Winner Determination Algorithm A_{winner}

Input: $\mathcal{J}_{\hat{T}_g}$, $\hat{T}_g$, K;
Output: $\mathcal{S}$, $\mathcal{P}$, $cost$;
1: Initialize $\mathcal{S} = \mathcal{P} = \mathcal{C} = \mathcal{G} = \varnothing$, $cost = 0$, $\gamma_t^S = 0, \forall t$;
2: **while** $R(\mathcal{S}) < K\hat{T}_g$ **do**
3: Sort $\hat{\mathcal{T}}_g$ global iterations according to γ_t^S in nondecreasing order; Save the order as $\tilde{T}$;
4: **for** all bids in $\mathcal{J}_{\hat{T}_g}$ **do**
5: Select the top c_{ij} global iterations in $\tilde{T}$ and within time period $[a_{ij}, d_{ij}]$ to form the representative schedule l_{ij};
6: Compute $R_{il_{ij}}(\mathcal{S})$; Save/update schedule (i, l_{ij}) in sets $\mathcal{C}$ and $\mathcal{G}$;
7: **end for**
8: $(i^*, l^*) = \arg\min_{(i,l_{ij})\in\mathcal{C}} \frac{\rho_{il_{ij}}}{R_{il_{ij}}(\mathcal{S})}$;
9: $z_{i^*l^*} = 1; \phi(t, l^{i^*}) = \frac{\rho_{i^*l^*}}{R_{i^*l^*}(\mathcal{S})}, \forall t \in \mathcal{F}_{i^*l^*}$;
10: $p_{i^*} = A_{payment}(\mathcal{C}, (i^*, l^*), R_{il_{ij}}(\mathcal{S}))$;
11: $(i^\#, l^\#) = \arg\min_{(i,l_{ij})\in\mathcal{G}} \frac{\rho_{il_{ij}}}{R_{il_{ij}}(\mathcal{S})}$;
12: $\phi(t, l^{i^\#})' = \frac{\rho_{i^\#l^\#}}{R_{i^\#l^\#}(\mathcal{S})}, \forall t \in \mathcal{F}_{i^\#l^\#}$;
13: $\mathcal{C} = \mathcal{C} \backslash (\bigcup_l (i^*, l))$;
14: $\mathcal{S} = \mathcal{S} \bigcup (i^*, l^*)$; $\mathcal{G} = \mathcal{G} \backslash (i^*, l^*)$;
15: **end while**
16: $\psi_{\max}^t = \max_{t\in[a_{ij},d_{ij}]}\{\rho_{il_{ij}}\}, \forall t \in \hat{\mathcal{T}}_g$;
17: $\psi_{\min}^t = \min_l\{\{\phi(t,l)\} \bigcup \{\phi(t, l^{i^\#})'\}\}, \forall t \in \hat{\mathcal{T}}_g$;
18: $\omega_t = \psi_{\max}^t / \psi_{\min}^t, \forall t \in \hat{\mathcal{T}}_g$; $\omega = \max_{t\in\hat{\mathcal{T}}_g} \omega_t$;
19: $\eta_\phi(t) = \max_l\{\phi(t,l)\}$, $g(t) = \eta_\phi(t)/(H_{\hat{T}_g}\omega), \forall t \in \hat{\mathcal{T}}_g$;
20: **for** all $z_{il} == 1$ **do**
21: $\lambda_{il_{ij}} = \sum_{t:t\in\mathcal{F}_{il}} (\eta_\phi(t) - \phi(t,l))/(H_{\hat{T}_g}\omega)$;
22: Save p_i to $\mathcal{P}$; $cost = cost + \rho_{il_{ij}}$;
23: **end for**

Average Cost. Let $\mathcal{S} = \{(i_1, l_1), (i_2, l_2), \ldots\}$ be a set where (i_1, l_1) is client i_1's l_1-th schedule. $\gamma_t^{\mathcal{S}} = \sum_{(i,l)\in\mathcal{S}: y_i(t)\in l} 1$ denotes the number of clients which are scheduled at the t-th global iteration in set $\mathcal{S}$. The utility of set $\mathcal{S}$ is its valid contribution, which is defined as $R(\mathcal{S}) = \sum_{t\in\hat{\mathcal{T}}_g} \min(\gamma_t^{\mathcal{S}}, K)$. The increased utility of adding client i's l-th schedule to $\mathcal{S}$ is:

$$\begin{aligned} R_{il}(\mathcal{S}) &= R(\mathcal{S} \bigcup (i,l)) - R(\mathcal{S}) \\ &= \sum_{t\in\hat{\mathcal{T}}_g} (\min(\gamma_t^{\mathcal{S}\bigcup(i,l)}, K) - \min(\gamma_t^{\mathcal{S}}, K)) \end{aligned} \tag{15}$$

The *average cost* of schedule l is $\frac{\rho_{il}}{R_{il}(\mathcal{S})}$. At the beginning, $\mathcal{S}$ is an empty set. In each iteration, the schedule with the minimum average cost is added to set $\mathcal{S}$, until there are enough participants. Although the number of feasible schedules for client i's j-th bid is up to $\binom{d_{ij}-a_{ij}}{c_{ij}}$ for each bid, we only need to consider one representative schedule which generates the maximum utility. Let l_{ij} denote the representative schedule for client i's j-th bid. l_{ij} consists of c_{ij} global iterations which have the smallest γ_t^S within the time period $[a_{ij}, d_{ij}]$.

Algorithm Details. The winner determination algorithm A_{winner} is shown in Algorithm 2. Here $\mathcal{C}$ is a candidate set which records representative schedules for selection during each iteration. $\mathcal{G}$ is a grand set which records unselected representative schedules. Let $\mathcal{F}_{il}$ be the set that stores the current *available* global iterations in client i's l-th schedule. Line 1 initializes sets and variables. Note that the default value of all primal and dual variables is zero. In lines 2-15, the *while* loop selects schedules iteratively until all global iterations have K participants. Lines 3-7 compute the representative schedule l_{ij} for each bid. Line 8 selects the schedule (i^*, l^*) with the smallest average cost. Then, the corresponding variable $z_{i^*l^*}$ is updated to 1, and its average cost $\phi(t, l^{i^*})$ is recorded in line 9. Line 10 calls the subroutine $A_{payment}$ to calculate the payment for each new selected schedule (i^*, l^*). Lines 11-12 record auxiliary variables for updating dual variables. Lines 13-14 update sets $\mathcal{C}$, $\mathcal{S}$ and $\mathcal{G}$. To satisfy constraint (14 b), set $\mathcal{C}$ removes all remaining schedules of client i^*. Then, the winner set $\mathcal{S}$ adds the new selected schedule (i^*, l^*), and set $\mathcal{G}$ excludes it. In order to bound the approximation ratio, A_{winner} updates dual variables. The value of dual variable $g(t)$ is calculated in lines 16-19, where $H_{\hat{T}_g} = \sum_{t=1}^{T_g} \frac{1}{t}$ is a harmonic number. Finally, lines 20-23 compute value of dual variable $\lambda_{il_{ij}}$ for each selected schedule and save winners' information.

Example. We illustrate the process of A_{winner} by a simple example. Suppose $\hat{T}_g = 3$ and $K = 1$. For any client $i \in \mathcal{I}$, it only submits one bid with a form of $B_{i1}(b_{i1}, [a_{i1}, d_{i1}], c_{i1})$. There are three qualified bids in set $\mathcal{J}_3$: $B_{11}(\$2, [1,2], 1)$, $B_{21}(\$6, [2,3], 2)$, $B_{31}(\$5, [1,3], 2)$. First, A_{winner} initializes $R(\mathcal{S}) = 0$ and $\gamma_t^{\mathcal{S}} = 0, \forall t \in [1,2,3]$.

- In the first iteration, the candidate set $\mathcal{C}$ includes three representative schedules: $l_{11} = \{1\}, l_{21} = \{2,3\}, l_{31} = \{1,2\}$. Since $R(\mathcal{S}) < 3$, A_{winner} computes $\frac{\rho_{11}}{R_{11}(\mathcal{S})} = 2$, $\frac{\rho_{21}}{R_{21}(\mathcal{S})} = 3$, $\frac{\rho_{31}}{R_{31}(\mathcal{S})} = 2.5$. $(1, l_{11})$ is selected since it has the minimum average cost. Its corresponding payment is calculated as $p_1 = R_{11}(\mathcal{S}) \cdot \frac{\rho_{31}}{R_{31}(\mathcal{S})} = 2.5$. Next, A_{winner} updates $\mathcal{S} = \{(1, l_{11})\}$ and $R(\mathcal{S}) = 1$, and then removes l_{11} from set $\mathcal{C}$.
- In the second iteration, the candidate set $\mathcal{C}$ contains two representative schedules: $l_{21} = \{2,3\}$, $l_{31} = \{2,3\}$. Because $R(\mathcal{S}) < 3$, A_{winner} computes $\frac{\rho_{21}}{R_{21}(\mathcal{S})} = 3$, $\frac{\rho_{31}}{R_{31}(\mathcal{S})} = 2.5$. $(3, l_{31})$ is selected. The corresponding payment is $p_3 = R_{31}(\mathcal{S}) \cdot \frac{\rho_{21}}{R_{21}(\mathcal{S})} = 6$. So $\mathcal{S} = \{(1, l_{11}), (3, l_{31})\}$ and $R(\mathcal{S}) = 3$. l_{31} is removed from set $\mathcal{C}$. The while loop in A_{winner} terminates since $R(\mathcal{S}) = K\hat{T}_g = 3$ now.

5.3 Payment Design

We next discuss how to calculate the payment for winners. The basic idea is to calculate the payment based on the critical bid, *i.e.,* the schedule which has the second smallest average cost. (Please see Theorem 1 for details). $A_{payment}$ is shown in Alg. 3. Line 1 finds the critical bid $(i,' l')$ and line 2 calculates the payment for each new selected schedule (i^*, l^*) based on the critical value rule [20], [21].

Algorithm 3. Payment Algorithm $A_{payment}$

Input: $\mathcal{C}$, (i^*, l^*), $R_{il_{ij}}(\mathcal{S})$;
Output: p_{i^*};
1: $(i', l') = \arg\min_{(i,l_{ij})\in\mathcal{C}:(i,l_{ij})\neq(i^*,l^*)} \frac{\rho_{il_{ij}}}{R_{il_{ij}}(\mathcal{S})}$;
2: $p_{i^*} = R_{i^*l^*}(\mathcal{S}) \cdot \frac{\rho_{i'l'}}{R_{i'l'}(\mathcal{S})}$;
3: return p_{i^*};

6 Theoretical Analysis

In this section, we analyze the property of A_{FL} in terms of truthfulness, individual rationality, correctness, time complexity and approximation ratio.

6.1 Truthfulness and Individual Rationality

Lemma 1. *A_{FL} is schedule-monotonic,* i.e., $\forall i \in \mathcal{I}, \forall l, \tilde{l} \in \mathcal{L}_i$, *if* $\rho_{i\tilde{l}} < \rho_{il}$ *and* $R_{i\tilde{l}}(\mathcal{S}) = R_{il}(\mathcal{S})$, $z_{il} = 1$ *implies* $z_{i\tilde{l}} = 1$.

Proof. When client i's schedule l was selected, then it has the lowest average cost $\frac{\rho_{il}}{R_{il}(\mathcal{S})}$ in the current iteration. If client i changes its cost to a smaller one $\rho_{i\tilde{l}}$ $(< \rho_{il})$ and others remain the same, $\frac{\rho_{i\tilde{l}}}{R_{il}(\mathcal{S})} < \frac{\rho_{il}}{R_{il}(\mathcal{S})}$ implies that this schedule will still be selected in the current iteration by the greedy algorithm A_{winner}. Hence, Lemma 1 holds. □

Lemma 2. *The payment to all selected schedules are critical,* i.e., *suppose that a selected schedule ($z_{i^*l^*} = 1$) has a bidding price $\tilde{\rho}_{i^*l^*}$ $(\neq \rho_{i^*l^*})$, then this schedule will win if $\tilde{\rho}_{i^*l^*} \leq p_{i^*}$, and will fail otherwise.*

Proof. According to $A_{paymemt}$, schedule $(i,' l')$ has the second smallest average cost in set $\mathcal{C}$ at the current iteration. Then, the payment $p_{i^*} = R_{i^*l^*}(\mathcal{S}) \cdot \frac{\rho_{i'l'}}{R_{i'l'}(\mathcal{S})}$ ensures that $\frac{\tilde{\rho}_{i^*l^*}}{R_{i^*l^*}(\mathcal{S})} \leq \frac{\rho_{i'l'}}{R_{i'l'}(\mathcal{S})}$ when $\tilde{\rho}_{i^*l^*} \leq p_{i^*}$ and $\frac{\tilde{\rho}_{i^*l^*}}{R_{i^*l^*}(\mathcal{S})} \geq \frac{\rho_{i'l'}}{R_{i'l'}(\mathcal{S})}$ when $\tilde{\rho}_{i^*l^*} \geq p_{i^*}$. Consequently, each winning schedule is paid with a critical value. □

Theorem 1. *A_{FL} is a* truthful *auction.*

Proof. (*Truthfulness in bidding price b_{ij}*): The Myerson's theorem [20], [21] shows that a reverse auction is truthful in bidding price if the following conditions are satisfied: (i) the result of auction (z_{il}) is monotonically non-decreasing with the decrease of bidding price ρ_{il}; and (ii) the payment of each selected schedule is calculated based on the critical value. Combining Lemma 1 and 2, we finish this part of proof.

(*Truthfulness in local accuracy θ_{ij}*): We first prove that client i will not report a smaller local accuracy θ'_{ij} than its true local accuracy θ_{ij}. A smaller local accuracy leads to a longer computation time, which will risk failing to satisfy the time limitation of one single communication round, *i.e.*, constraint (7). Even if client i submits a smaller local accuracy and it is selected by the FL platform, client i cannot achieve the local accuracy that it claimed. Therefore, the FL platform will refuse to pay when this happened.

If client i bids with a larger local accuracy, it would reduce the probability of being accepted by the FL platform. This is because the larger local accuracy may not satisfy the accuracy requirement of FL job, *i.e.*, constraint (5). Thus, clients will not misreport the local accuracy of their bids.

(*Truthfulness in available time period $[a_{ij}, d_{ij}]$ and the number of participation rounds c_{ij}*): If client i reports a longer available time period and it is accepted by the FL platform, client i may not be able to participate in some rounds due to its actual schedule. Hence, the FL platform will refuse to pay when this happened. If client i claims a shorter available time period, the average cost of that bid may increase and it will further reduce the probability of acceptance. The reason is that the average cost is calculated based on the increased utility of adding one schedule, *i.e.*, $R_{il}(\mathcal{S})$, and the value of $R_{il}(\mathcal{S})$ may reduce because a shorter available time period will narrow down the range that the schedule l can select. In summary, there is no incentive to misreport the available time period.

Similarly, clients who submit a larger number of participation rounds would not get the payment from the FL platform, since they actually can not provide the service as they claimed. On the other hand, a smaller number of participation rounds submitted by clients results in a higher average cost, reducing the likelihood of acceptance. Therefore, clients will not misreport the number of participation rounds.

In conclusion, A_{FL} is a truthful auction. □

Theorem 2. *A_{FL} achieves individual rationality.*

Proof. The payment of selected schedule (i^*, l^*) is based on the critical value. It is clear that $(i,' l')$'s average cost will be no less than (i^*, l^*)'s, *i.e.*, $\frac{\rho_{i^*l^*}}{R_{i^*l^*}(\mathcal{S})} \leq \frac{\rho_{i'l'}}{R_{i'l'}(\mathcal{S})}$. Then, we have $\rho_{i^*l^*} \leq R_{i^*l^*}(\mathcal{S}) \cdot \frac{\rho_{i'l'}}{R_{i'l'}(\mathcal{S})} = p_{i^*}$. Furthermore, Theorem 1 ensures truthfulness, *i.e.*, $v_{i^*l^*} = \rho_{i^*l^*}$. Therefore, each client's utility, $u_{i^*l^*} = p_{i^*} - v_{i^*l^*} \geq 0$, is always non-negative. □

6.2 Correctness and Time Complexity

Lemma 3. *A_{winner} produces a feasible solution to ILP (13) and LP (14).*

Proof. We first prove that A_{winner} in Alg. 2 returns a feasible solution to ILP (13). If there exists enough clients, Alg. 2 has at least one feasible solution and it can terminate either before or when set $\mathcal{C} = \varnothing$. When Alg. 2 terminates, the ending condition of the *while* loop can guarantee that constraint (13 a) is satisfied. Then, constraint (13 b) is not violated since line 13 in Alg. 2 removes all remaining schedules corresponding to bids of client i^* from set $\mathcal{C}$. Constraint (13 c) holds because the default value of $z_{il_{ij}}$ is zero and it is updated to 1 only when client i's l_{ij}-th schedule is selected. In conclusion, A_{winner} generates a feasible solution to ILP (13). □

We next prove that A_{winner} in Alg. 2 returns a feasible solution to LP (14) in two cases.

Case 1: If client i's l-th schedule is not selected by Alg. 2. we first sort all global iterations in client i's l-th schedule in non-decreasing according of $\gamma_t^{\mathcal{S}}$, and denote it as $\tilde{t} = \{t_1, t_2, \ldots, t_{|c_{ij}|}\}$. Let t_k be the k-th global iteration in $\tilde{t}$. If t_k is *available* (*i.e.* $\gamma_{t_k}^{\mathcal{S}} \leq K$), schedule l has at least k *available* global iterations. Thus, the average cost for client i's l-th schedule to cover *available* global iterations is no larger than $\frac{\rho_{il}}{k}$. Note that $\psi_{\max}^t$ is the maximum bidding pricing in t-th global iteration, and $\psi_{\min}^t$ is the minimum average cost. Therefore, the cost of t_k-th global iteration $\eta_\phi(t_k)$ is no larger than $\frac{\rho_{il}}{k}\frac{\psi_{\max}^{t_k}}{\psi_{\min}^{t_k}}$, when t_k-th global iteration has K participants. Then

$$\begin{aligned}
&\sum_{t:y_i(t)\in l} g(t) - \lambda_{il} - q_i = \sum_{t:y_i(t)\in l} g(t) \\
&= \frac{1}{H_{\hat{T}_g}\omega} \sum_{t:y_i(t)\in l} \eta_\phi(t) \leq \frac{\rho_{il}}{H_{\hat{T}_g}\omega} \sum_{k=1}^{|c_{ij}|} \frac{1}{k} \frac{\psi_{\max}^{t_k}}{\psi_{\min}^{t_k}} \\
&= \frac{\rho_{il}}{H_{\hat{T}_g}\omega} H_{|c_{ij}|}\omega \leq \frac{\rho_{il}}{H_{\hat{T}_g}\omega} H_{\hat{T}_g}\omega = \rho_{il}.
\end{aligned}$$

Therefore, constraint (14 a) can be satisfied when client i's l-th schedule is not selected.

Case 2: If client i's l-th schedule is selected by Alg. 2. Then, we have

$$\begin{aligned}
&\sum_{t:y_i(t)\in l} g(t) - \lambda_{il} - q_i = \sum_{t:y_i(t)\in l} g(t) - \lambda_{il} \\
&= \frac{1}{H_{\hat{T}_g}\omega} \left(\sum_{t:t\in l\backslash\mathcal{F}_{il}} \eta_\phi(t) + \sum_{t\in\mathcal{F}_{il}} \phi(t,l) \right) \\
&= \frac{1}{H_{\hat{T}_g}\omega} \left(\sum_{t:t\in l\backslash\mathcal{F}_{il}} \eta_\phi(t) + \rho_{il} \right) \\
&\leq \frac{\rho_{il}}{H_{\hat{T}_g}} \left(\sum_{k=1}^{|l\backslash\mathcal{F}_{il}|} \frac{1}{k+|\mathcal{F}_{il}|} + \frac{1}{\omega} \right) \\
&\leq \rho_{il} \left(\frac{H_{|c_{ij}|} - H_{|\mathcal{F}_{il}|} + 1}{H_{\hat{T}_g}} \right) \leq \rho_{il}.
\end{aligned}$$

Recall that $\mathcal{F}_{il}$ represents a set involves those global iterations within schedule l that still *available*. And $l\backslash\mathcal{F}_{il}$ denotes the set of *non-available* global iterations in schedule l. Similarly, we order all global iterations in set $l\backslash\mathcal{F}_{il}$ as $\{t_1, t_2, \ldots, t_{|l\backslash\mathcal{F}_{il}|}\}$. If t_k-th global iteration is just *non-available* after client i's l-th schedule is selected, its cost $\eta_\phi(t)$ should be at most $\frac{\rho_{il}}{k+|\mathcal{F}_{il}|}\frac{\psi_{\max}^{t_k}}{\psi_{\min}^{t_k}}$. Hence, constraint (14 a) can also be satisfied.

In summary, A_{winner} generates a feasible solution to ILP (13) and LP (14).

Lemma 4. *The running time of A_{winner} is $O(I\hat{T}_g\ (\log(\hat{T}_g) + IJ))$.*

Proof. Line 1 of A_{winner} in Alg. 2 first defines and initializes variables in $O(\hat{T}_g)$ steps. The *while* loop (lines 2-15) runs at most I iterations since the number of clients can be selected is at most I. Sorting all global iterations within $\hat{\mathcal{T}}_g$ needs at least $O(\hat{T}_g \log(\hat{T}_g))$ steps. The inner *for* loop in lines 4-7 selects and updates the representative schedule for each bid, which can be calculated in $O(IJ\hat{T}_g)$ steps. To find a schedule (i^*, l^*) or $(i^\#, l^\#)$, we need to search all schedules within the corresponding set, which takes $O(IJ)$ steps. Then, computing the payment in Alg. 3, updating three related sets $\mathcal{C}, \mathcal{G}$ and $\mathcal{S}$, and recording its cost can be done within $O(IJ\hat{T}_g)$ steps. Consequently, the running time of the *while* loop in Alg. 2 is $O(I\hat{T}_g(\log(\hat{T}_g) + IJ))$. Lines 16-19 calculate dual variable $g(t)$, which take $O(IJ\hat{T}_g)$ steps. The second *for* loop takes $O(IJ)$ steps to calculate dual variable λ_{il}, save corresponding payment p_i, and record the total cost. In conclusion, the time complexity of A_{winner} is $O(I\hat{T}_g(\log(\hat{T}_g) + IJ))$. □

Theorem 3. *A_{FL} produces a feasible solution to ILP (12) in polynomial time.*

Proof. We first prove the time complexity of A_{FL}. Lines 1-2, and 12-14 of A_{FL} can be finished within $O(IJ)$ steps. Line 6 in Alg. 1 needs to search all bids, which takes $O(IJ)$ steps. According to Lemma 4, we know that the time complexity of A_{winner} is $O(I\hat{T}_g(\log(\hat{T}_g) + IJ))$. Therefore, the *for* loop in Alg. 1 which includes Alg. 2 can be done within $O(IT^2(\log(T) + IJ))$ steps. In summary, the time complexity of A_{FL} is $O(IT^2(\log(T) + IJ))$. □

Next, we discuss the correctness of A_{FL}. Constraint (12 h) holds since we enumerate $\hat{T}_g$ in the *for* loop (lines 4-11). Before solving ILP (13), we pick those bids which satisfy constraint (5) and (7) at different fixed $\hat{T}_g$ and form a qualified set $\mathcal{J}_{\hat{T}_g}$ for ILP (13) (line 6). Therefore, constraint (5) and (7) hold. Finally, the correctness of A_{FL} can be guaranteed by combining Lemma 3.

In conclusion, A_{FL} produces a feasible solution to ILP (12) in polynomial time.

6.3 Approximation Ratio

Here, we prove that the theoretical approximation ratio of A_{FL} is $H_{\hat{T}_g^*}\omega$. Furthermore, the value of $H_{\hat{T}_g^*}\omega$ is around 1.1, which will be verified in our simulations in Section 8.

Definition 4. *(Approximation Ratio): The approximation ratio of an algorithm A for a minimization problem is the upper bound ratio of the objective value of the solution found by A over the objective value returned by an optimal algorithm.*

Lemma 5. *Let P and D be the objective values of the primal problem (13) and the dual problem (14) returned by A_{winner}. $\tau D \geq P$ with $\tau = H_{\hat{T}_g}\omega$, where $H_{\hat{T}_g} = \sum_{t=1}^{\hat{T}_g} \frac{1}{t}$ and ω is defined in line 18 of A_{winner}. A_{winner} is a τ-approximation algorithm.*

Proof. The objective value of dual problem (14) is

$$\begin{aligned}
D &= \frac{1}{H_{\hat{T}_g}\omega} \sum_{t\in\hat{\mathcal{T}}_g} K\eta_\phi(t) - \frac{1}{H_{\hat{T}_g}\omega} \sum_{(i,l)} \sum_{t:t\in\mathcal{F}_{il}} (\eta_\phi(t) - \phi(t,l)) \\
&= \frac{1}{H_{\hat{T}_g}\omega} \left(\sum_{t\in\hat{\mathcal{T}}_g} K\eta_\phi(t) - \sum_{t:t\in\mathcal{F}_{il}} \sum_{(i,l)} \eta_\phi(t) \right) \\
&\quad + \frac{1}{H_{\hat{T}_g}\omega} \sum_{(i,l)} \sum_{t:t\in\mathcal{F}_{il}} \phi(t,l) \\
&\geq \frac{1}{H_{\hat{T}_g}\omega} \sum_{(i,l)} \sum_{t\in\hat{\mathcal{T}}_g} \phi(t,l).
\end{aligned}$$

□

For each global iteration in set $\mathcal{F}_{il}$, the number of selected clients is no larger than K. Hence, the first term of the second equality is larger than 0. Meanwhile, $\phi(t,l)$ is assigned

a value only when t-th global iteration belongs to set $\mathcal{F}_{il}$. Therefore, it is rational to extend the range of t in the term $1/(H_{\hat{T}_g}\omega)\sum_{(i,l)}\sum_{t:t\in\mathcal{F}_{il}}\phi(t,l)$ from $t:t\in\mathcal{F}_{il}$ to $t\in\hat{T}_g$.

Then, the objective value of primal problem (13) is

$$P=\sum_{(i,l)\in\mathcal{S}}\rho_{il}=\sum_{(i,l)}\sum_{t\in\hat{T}_g}\phi(t,l).$$

The above equation holds since when client i's l-th schedule is selected by A_{winner}, ρ_{il} is evenly distributed to variables $\phi(t,l)$, *i.e.*, all global iterations in $\mathcal{F}_{il}$.

Obviously, $H_{\hat{T}_g}\omega\cdot D\geq P$. Let P^* denote the optimal objective value of ILP (13). We have $P^*\geq D$ according to LP duality [45]. Consequently, $P/P^*\leq P/D\leq H_{\hat{T}_g}\omega=\tau$. Therefore, the approximation ratio of A_{winner} is τ.

Theorem 4. *The approximation ratio of A_{FL} is τ^* where $\tau^*=H_{\hat{T}_g^*}\omega$.*

Proof. Suppose that the optimal number of global iterations is $\hat{T}_g^*$. Let C^* denote the optimal social cost. Let $C^\#$ be the social cost returned by A_{FL}. C denotes the cost of corresponding solution returned by A_{winner} under fixed $\hat{T}_g^*$. Then, we have $C^\#\leq C$. Hence, $C^\#/C^*\leq C/C^*\leq H_{\hat{T}_g^*}\omega=\tau^*$. Therefore, we can conclude that A_{FL} in Alg. 1 is a τ^*-approximation algorithm. □

7 Extension to Multiple Jobs Scenario

In this section, we discuss one realistic FL scenario where there are multiple FL jobs with corresponding budget functions. We reformulate the social cost minimization problem in Section 7.1. Then we present our improved algorithm framework A_{FL-M} in Section 7.2 and further analyze its properties in Section 7.3.

7.1 System Model

In practice, the cloud server may receive requests of training jobs (models) for different purposes, such as traffic crowdsensing [46], location prediction [47], air pollution monitoring [48]. In this section, we will discuss how to select participated clients in the multiple jobs' scenario. Assume that there are M FL jobs. Different from the single job scenario, FL platform needs to further determine how to select proper participated clients for each FL job to minimize the social cost. To be more practical, we further introduce the concept of budget for FL jobs. Note that one FL job's owner hopes that his job can complete as soon as possible, which can be characterized by the budget. That is, the budget of job decreases monotonically with its completion time, *i.e.*, the number of global iterations T_{gm}. In this regard, we assume that FL job m is associated with one specific budget function $F_{Bm}(T_{gm})$, which decreases with T_{gm} monotonically. In summary, we have to deal with the case of processing multiple FL jobs with different budgets. Here, each client can submit bids to multiple jobs according to its preference, but it can only participate in one job due to its limited battery capacity. We assume that client i submits up to J bids for each FL job m, and client i's j-th bid for FL job m (B_{ijm}) can be expressed as a tuple:

$$B_{ijm}=\{b_{ijm},\theta_{ijm},[a_{ijm},d_{ijm}],c_{ijm}\}_{\forall j\in\mathcal{J},\forall m\in\mathcal{M}},\tag{16}$$

where b_{ijm} is the "claimed" cost that client i wants to charge for the service. θ_{ijm} is the local accuracy. $[a_{ijm},d_{ijm}]$ is the available time period within $\mathcal{T}$, which starts and ends at a_{ijm}-th and d_{ijm}-th global iteration.

Problem Reformulation. Under truthful bidding ($b_{ijm}=v_{ijm}$) and the budget function, the social cost minimization problem can be reformulated into the following integer linear program (ILP):

$$\text{minimize}\quad\sum_{m\in\mathcal{M}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}b_{ijm}x_{ijm}\tag{17}$$

subject to:

$$\sum_{i\in\mathcal{I}}y_{im}(t)\geq K_m,\quad\forall t\in\mathcal{T}_{gm},\forall m\in\mathcal{M},\tag{17a}$$

$$T_{gm}\geq\frac{1}{1-\theta_{ijm}x_{ijm}},\quad\forall i\in\mathcal{I},\forall j\in\mathcal{J},\forall m\in\mathcal{M},\tag{17b}$$

$$\sum_{t\in\mathcal{T}_{gm}}y_{im}(t)=\sum_{j\in\mathcal{J}}c_{ijm}x_{ijm},\quad\forall i\in\mathcal{I},\forall m\in\mathcal{M},\tag{17c}$$

$$x_{ijm}\cdot(T_{lm}(\theta_{ijm})t_{im}^{cmp}+t_{im}^{com})\leq t_{\max},\quad\forall i\in\mathcal{I},\forall j\in\mathcal{J},\forall m\in\mathcal{M},\tag{17d}$$

$$y_{im}(t)=1\text{ only if }x_{ijm}=1,\quad t\in[a_{ijm},d_{ijm}],\quad\forall i\in\mathcal{I},\forall j\in\mathcal{J},\forall m\in\mathcal{M},\tag{17e}$$

$$\sum_{m\in\mathcal{M}}\sum_{j\in\mathcal{J}}x_{ijm}\leq 1,\quad\forall i\in\mathcal{I},\tag{17f}$$

$$\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}p_{im}x_{ijm}\leq F_{Bm}(T_{gm}),\quad\forall m\in\mathcal{M},\tag{17g}$$

$$x_{ijm}b_{ijm}\leq p_{im},\quad\forall i\in\mathcal{I},\forall j\in\mathcal{J},\forall m\in\mathcal{M},\tag{17h}$$

$$x_{ijm},y_{im}(t)\in\{0,1\},\quad\forall i\in\mathcal{I},\forall j\in\mathcal{J},\forall t\in\mathcal{T}_{gm},\forall m\in\mathcal{M},\tag{17i}$$

$$T_{gm}\in\{1,2,\ldots,T\},\quad\forall m\in\mathcal{M}.\tag{17j}$$

Constraints (17 a)-(17 f) are the same as constraints (4)-(9). Especially, constraint (17 f) indicates each client can only be accepted one bid even in the multiple job scenario due to its limited battery capacity. Constraint (17 g) ensures that the total payment of the selected clients will not exceed the budget of job m. Constraint (17 h) limits the value of p_{im}. Note that we add one extra subscript m to all related parameters to identify FL job m. The value of budget is a constant when $\hat{T}_{gm}$ is fixed according to the function $F_{Bm}(T_{gm})$. Therefore, the reformulation of WDP (17) with a fixed $\hat{T}_{gm}$ is consistent with the ILP (13).

7.2 Algorithm Design

To address ILP (17), we present a revised auction framework A_{FL-M}, as shown in Alg. 4 to select bids and corresponding schedules for jobs. A_{FL-M} includes two subroutines: i) an algorithm to determine the value of T_{gm}^* to achieve the lowest social cost for each job; ii) a winner determination algorithm named $A_{winner-M}$, which aims to select proper clients and corresponding schedules via a greedy fashion for each job.

7.2.1 Auction Framework A_{FL-M}

We first invoke Alg. 5 to derive the value of T_{gm}^* and the candidate set $\mathcal{J}_{\hat{T}_{gm}}$ for each job in parallel (line 2). However, there are specific cases that some clients' bids are selected by

several different FL jobs at the same time. Recall that each client can only be accepted at most one bid. To address those extreme cases, from the prospective of clients, we consider rescheduling clients for FL jobs via a greedy fashion. That is, if one client's bid has been selected by multiple jobs, we will assign this client for the job with smaller rescheduling cost. The extra cost caused by rescheduling one client[4] (*i.e.*, rescheduling cost $recost$) for job m equals the cost difference between the pre-selected bid and the critical client i' bid, *i.e.*, $b_{i'jm} - b_{ijm}$. Let Υ_i denote a tuple, which records the job m that client i is selected for and the corresponding replace cost, *i.e.*, $\Upsilon_i = \{m, recost_{il_{ijm}}\}$. In Alg. 4, lines 3-25 select clients for all jobs until there are enough clients selected for each job to participate in the training process. Especially, line 4 uses $A_{winner-M}$ to select one client i^* who can participate in one specific time period for each job, and calculate its $recost_{i^*l^*_{ijm}}$. Note that it is possible that some FL jobs cannot be completed due to the limited number of clients. Therefore, we intend to preferentially allocate clients for those jobs that achieve T^*_{gm}, since it can effectively maximize the total social welfare, *i.e.*, social cost. Lines 5-13 choose the job with smaller rescheduling cost for client i, and further update Υ_i. Other jobs which have selected client i through Alg. 5 will remove the corresponding schedule of client i from the winner set $\mathcal{S}_m$. Lines 15-23 judge whether job m is valuable to train. Let $\hat{P}_m(\hat{T}_{gm})$ denote one job's utility, which equals the difference between job's budget with $\hat{T}_{gm}$ and the overall payment, *i.e.*, $\hat{P}_m(\hat{T}_{gm}) = F_{Bm}(\hat{T}_{gm}) - \sum_{i\in\mathcal{I}} p_{im}$. In line 15, if $\hat{P}_m(T^*_{gm}) < 0$, the FL platform will reject job m and remove it from the job set $\mathcal{M}$. Lines 16-20 update $flag_m$ with 1 if the job's training requirement is satisfied, and $flag_m = 0$ otherwise.

7.2.2 Subroutine for Finding T^*_{gm}

To reduce the time complexity of traversing $\hat{T}_{gm}$, we first build a curve of the social cost by sampling a number of global iterations $\hat{T}_{gm}$ with a fixed interval. Then we can observe a specific value $\hat{T}'_{gm}$ with the smallest social cost among samples. Note that $\hat{T}'_{gm}$ we obtained through sampling is close to the optimal T^*_{gm}, which has the smallest social cost. To avoid falling into the local optimum, we adopt the method of "jumping out" [49]. That is, we can search in other directions to jump out the local optimum point by adding a random perturbation through Cauchy mutation. To achieve a more precise result, we further apply binary search to a range nearby $\hat{T}'_{gm}$. The modified algorithm Alg. 5 is shown as below.

In Alg. 5, lines 3-13 sample α global iterations $\hat{T}_{gm}$ in the range $[T^m_0, T]$, with a fixed interval between them. Note that τ represents the fixed interval, which defined as $\tau = \lceil (T - T^m_0)/\alpha \rceil$. Here α denotes the number of samples. Lines 6-8 use Alg. 2 to calculate one possible solution for each WDP with fixed $\hat{T}_{gm}$. The condition in line 10 ($\hat{P}_m(\hat{T}_{gm}) \geq 0$) is needed to avoid violating constraint (17 g). After comparing all qualified samples, we finally get the value of $\hat{T}'_{gm}$, which has the smallest cost among them (Lines 10-12). Then we apply binary search to the range nearby $\hat{T}'_{gm}$, *i.e.*, the range $[T^-_m, T^+_m]$ (Lines 14-26). Note that parameter υ used for computing T^-_m is a constant, which can be adjusted according to the experiment's setting. Finally, we effectively find the near optimal T^*_{gm} with the smallest social cost $cost^*_m$.

Algorithm 4. FL Auction With Multi Jobs A_{FL-M}

Input: $M, T, K_m, B_{ijm}, \forall i, j, m$;
Output: $\sum_{m\in\mathcal{M}} cost_m$, $\{\mathcal{S}_m\}_{\forall m}$, $\{T^*_{gm}\}_{\forall m}$, $\{flag_m\}_{\forall m}$;

1: Initialize $\Upsilon_i, \forall i$; $T^*_{gm} = 0$, $flag_m = 0$, $cost_m = 0$, $R_m(\mathcal{S}_m) = 0$, $\mathcal{S}_m = \mathcal{P}_m = \varnothing, \forall m$;
2: Parallel calculate T^*_{gm} for each job m using Alg. 5;
3: **while** $\sum_{m\in\mathcal{M}} flag_m \leq M$ **do**
4: Parallel select clients for each uncompleted job m using Alg. 6;
5: **for** $m \in \mathcal{M}$ **do**
6: Find set Υ_i that involves job m;
7: **if** $\Upsilon_i \neq \varnothing$ **then**
8: Compare the $recost_{il_{ijm'}}$ of job m' in Υ_i with $recost_{il_{ijm}}$ of job m, save the job with smaller replace cost in Υ_i;
9: For job $\tilde{m}$ with larger $recost$, $\mathcal{S}_{\tilde{m}}\backslash(i, l_{ij\tilde{m}})$, $\mathcal{J}_{T^*_{g\tilde{m}}}\backslash(\bigcup_l (i, l_{ij\tilde{m}}))$, $z_{il_{ij\tilde{m}}} = 0$, remove $p_{i\tilde{m}}$ from $\mathcal{P}_{\tilde{m}}$, $cost_{\tilde{m}} = cost_{\tilde{m}} - \rho_{il_{ij\tilde{m}}}$, recompute $R_{\tilde{m}}(\mathcal{S}_{\tilde{m}})$;
10: **else**
11: Save $recost_{i^*l^*_{ijm}}$ to Υ_i;
12: **end if**
13: **end for**
14: **for** $m \in \mathcal{M}$ **do**
15: **if** $\hat{P}_m(T^*_{gm}) \geq 0$ **then**
16: **if** $R_m(\mathcal{S}_m) \geq K_m T^*_{gm}$ **then**
17: $flag_m = 1$;
18: **else**
19: $flag_m = 0$;
20: **end if**
21: **else**
22: Reject job m, $\mathcal{M} = \mathcal{M}\backslash m$, $flag_m = 0$;
23: **end if**
24: **end for**
25: **end while**
26: **for all** $x_{ijm} == 1, \forall x_{ijm} \in \mathcal{S}_m, \forall m$ **do**
27: Accept client i's j-th bid for job m and schedule client i according to $y_{im}(t) \in l_{ijm}$; Pay $p_{im} \in \mathcal{P}_m$ to client i;
28: **end for**

Note that $A_{winner-M}$ shown in Alg. 6 is similar to A_{winner}. Rather than selecting bids and corresponding schedules at one time, $A_{winner-M}$ just selects one bid based on the current schedule of job m, and calculates the corresponding replace cost.

7.3 Theoretical Analysis

In this section, we also analyze truthfulness, individual rationality, correctness, and time complexity of A_{FL-M}.

Theorem 5. *A_{FL-M} is a* truthful *auction and achieves individual rationality.*

Proof. We omit the proof here as it is similar to A_{FL}'s proof. □

Theorem 6. *A_{FL-M} produces a feasible solution to ILP (17) in polynomial time.*

4. In the remainder of this paper, we do not differentiate between a client and a bid unless otherwise stated since each client can only be accepted at most one bid.

Proof. We first analyze the time complexity of Alg. 5. Lines 1-2, and line 27 of Alg. 5 can be finished within $O(IJ)$ steps. Line 7 and line 18 in Alg. 5 need to search all bids, which takes $O(IJ)$ steps. According to Lemma. 4, the time complexity of A_{winner} is $O(I\hat{T}_g(\log(\hat{T}_g)+IJ))$. The *for* loop cycle $\alpha(\ll \hat{T}_{gm})$ times which is a constant. Therefore, the *for* loop in Alg. 5 which includes Alg. 2 can be done within $O(\alpha IT(\log(T)+IJ))$ steps. Then, the *while* loop in line 15-26 is a traditional binary search whose complexity is within $O(\log(\upsilon))$, which also can be regarded as a constant. So the *while* loop in Alg. 5 can be done within $O(T(\log(T)+IJ))$ steps. In summary, the time complexity of Alg. 5 is $O(IT(\log(T)+IJ))$. □

Algorithm 5. Subroutine for Finding T^*_{gm}

Input: T, K_m, B_{ijm}, $\forall i,j,m$;
Output: $\mathcal{J}_{T^*_{gm}}$, T^*_{gm};
1: Initialize $t_{ijm}=T_{lm}(\theta_{ijm})t_i^{cmp}+t_i^{com},\forall i,j$; $\mathcal{S}_m=\mathcal{P}_m=\mathcal{J}_{T^*_{gm}}=\varnothing$, $cost^*_m=\infty,\forall m$; $\mathcal{J}_{\hat{T}_{gm}}=\varnothing,\forall\hat{T}_{gm}$;
2: Find the minimum local accuracy $\theta^m_{\min}$ of all bids;
3: $T^m_0=\lfloor 1/(1-\theta^m_{\min})\rfloor$, $\tau=\lceil(T-T^m_0)/\alpha\rceil$, $\hat{T}_{gm}=T^m_0$;
4: **for** i = 1, 2, $\cdots$, α **do**
5: Given $\hat{T}_{gm}=\min\{\hat{T}_{gm}+\tau,T\}$;
6: $\theta^m_{\max}=\lceil 1-1/\hat{T}_{gm}\rceil$;
7: $\mathcal{J}_{\hat{T}_{gm}}=\{(i,j,m)_{\forall i,j,m}|\theta_{ijm}\le\ \theta^m_{\max}\&t_{ijm}\le t_{\max}\&a_{ijm}+c_{ijm}\le\hat{T}_{gm}\}$;
8: $(\mathcal{S}_m,\mathcal{P}_m,cost(\hat{T}_{gm}))$=$A_{winner}(\mathcal{J}_{\hat{T}_{gm}},\hat{T}_{gm},K_m)$;
9: Calculate $\hat{P}_m(\hat{T}_{gm})$;
10: **if** $cost(\hat{T}_{gm})<cost^*_m$ and $\hat{P}_m(\hat{T}_{gm})\ge 0$ **then**
11: $T'_{gm}=\hat{T}_{gm}$, $cost^*_m=cost(\hat{T}_{gm})$;
12: **end if**
13: **end for**
14: $T^-_m=\max(T^m_0,T'_{gm}-\upsilon)$, $T^+_m=T'_{gm}$;
15: **while** $T^-_m\le T^+_m$ **do**
16: $\hat{T}_{gm}=T^-_m+(T^+_m-T^-_m)/2$;
17: $\theta^m_{\max}=\lceil 1-1/\hat{T}_{gm}\rceil$;
18: $\mathcal{J}_{\hat{T}_{gm}}=\{(i,j,m)_{\forall i,j,m}|\theta_{ijm}\le\ \theta^m_{\max}\&t_{ijm}\le t_{\max}\&a_{ijm}+c_{ijm}\le\hat{T}_{gm}\}$;
19: $(\mathcal{S}_m,\mathcal{P}_m,cost(\hat{T}_{gm}))$=$A_{winner}(\mathcal{J}_{\hat{T}_{gm}},\hat{T}_{gm},K_m)$;
20: Calculate $\hat{P}_m(\hat{T}_{gm})$;
21: **if** $cost(\hat{T}_{gm})<cost^*_m$ and $\hat{P}_m(\hat{T}_{gm})\ge 0$ **then**
22: $T^*_{gm}=\hat{T}_{gm}$, $\mathcal{J}_{T^*_{gm}}=\mathcal{J}_{\hat{T}_{gm}}$, $cost^*_m=cost(\hat{T}_{gm})$, $T^+_m=\hat{T}_{gm}-1$;
23: **else**
24: $T^-_m=\hat{T}_{gm}+1$;
25: **end if**
26: **end while**
27: A_{FL} : Lines 12-14;

Then we analyze the time complexity of A_{FL-M}. In line 2 of Alg. 4, calculating the near optimal T^*_{gm} for each job parallelly needs $O(ITM(\log(T)+IJ))$ steps. To analyze the time complexity of the *while* loop (lines 3-25), we need to consider the worst case for one job, that is each selected client for one job is already preempted by others jobs. Note that this worse case can be regarded as excluding all other jobs' clients, and its time complexity is $O(I)$. Therefore, this *while* loop needs to execute at most $2IM$ steps since there are M jobs. In Alg. 4, lines 5-13 and 14-24 can be finished within $O(M)$ steps. Note that $A_{winner-M}$ can be regarded as one round of A_{winner}'s *while* loop. Hence, the time complexity of $A_{winner-M}$ is $O(\hat{T}_{gm}(\log(\hat{T}_{gm})+IJ))$. So the line 4 of Alg. 4 is executed at most $O(M\hat{T}_{gm}(\log(\hat{T}_{gm})+IJ))$ times. In summary, the time complexity of A_{FL-M} is $O(ITM^2(\log(T)+IJ))$.

Next, we prove the correctness of A_{FL-M}. Constraint (17 g) holds since we only identify the bids that satisfy the condition $\hat{P}_m(\hat{T}_{gm})\ge 0$ in lines 10-12 and lines 21-25 of Alg. 5. Meanwhile, lines 15-23 in Alg. 4 also guarantee constraint (17 g) not being violated. Constraint (17 h) is satisfied due to Theorem 5. Then we only sample the number of global iterations within the range $[T^m_0,T]$ in the *for* loop and the *while* loop, which can guarantee constraint (17 j). And constraint (17 b) and (17 d) both hold because of the qualified set $\mathcal{J}_{\hat{T}_{gm}}$ at different number of global iterations $\hat{T}_{gm}$ in line 7 and line 18 of Alg. 5. Moreover, the other constraints of ILP (17) can be satisfied in Alg. 6, which already discuss in Lemma 3. Then, constraint (17 f) is not violated since line 14 in Alg. 6 removes all remaining schedules of client i^* from set $\mathcal{C}_m$ and line 5-13 in Alg. 4 only select one schedule of client i^*. Constraint (17 a) holds because of the condition in lines 16-20 of Alg. 4.

Algorithm 6. Winner Determination Algorithm $A_{winner-M}$

Input: $\mathcal{J}_{T^*_{gm}}$, $R_m(\mathcal{S}_m)$, T^*_{gm}, $\mathcal{P}_m$, $cost_m$, $\mathcal{S}_m$, $\forall m$;
Output: $recost_{i^*l^*_{ijm}}$;
1: Initialize $\mathcal{C}_m=\mathcal{G}_m=\mathcal{J}_{T^*_{gm}}$, $\gamma_t^{S_m}=0,\forall t$;
2: A_{winner}: Lines 3-14;
3: Save p_{im} to $\mathcal{P}_m$, $cost_m=cost_m+\rho_{il_{ijm}}$, $\mathcal{J}_{T^*_{gm}}=\mathcal{C}_m$;
4: $recost_{i^*l^*_{ijm}}=\rho_{i'l'_{ijm}}-\rho_{i^*l^*_{ijm}}$;

In conclusion, Alg. 4 produces a feasible solution to ILP (17) in polynomial time.

8 Performance Evaluation

8.1 Evaluation Setup

System Settings. For fair comparison, we follow the similar setting in [4], [6]. By default, there are 1000 (I) clients and each client submits 5 (J) bids [4]. Assume that the maximum number of global iterations equals 50 and each global iteration needs 20 clients to train collaboratively, *i.e.*, $T=50$ and $K=20$ [4], [23]. t_i^{cmp} and t_i^{com} are randomly picked within the range of [5,10] and [10,15], respectively [6]. The local accuracy θ_{ij} of all bids are uniformly distributed in [0.3,0.8] [10], [8]. We calculate the number of local iterations $T_l(\theta_{ij})$ according to a simplified equation: $T_l(\theta_{ij})=\lceil 10(1-\theta_{ij})\rceil$ [4]. In our simulations, we do not consider the case that two available time periods overlap since they can be considered as one time period from the perspective of clients. Therefore, we select $2J$ non-repeated random numbers within the range $[1,T]$, and sort them in non-decreasing order to form J available time periods. The starting time (a_{ij}) and the ending time (d_{ij}) of each time period equal two adjacent random numbers in the order, respectively. The number of participation rounds (c_{ij}) is randomly generated

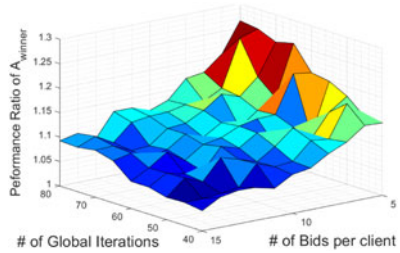

Fig. 3. Performance ratio of A_{winner}.

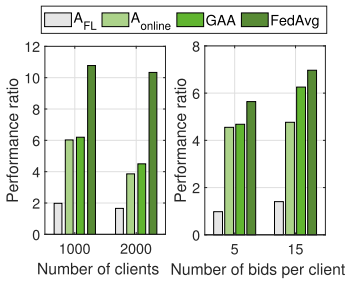

Fig. 4. Performance ratio of A_{FL}.

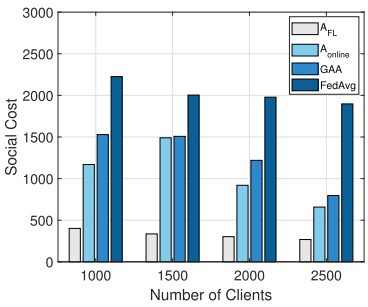

Fig. 5. Social cost under different I.

within the range [1, $d_{ij} - a_{ij}$]. Finally, the claimed cost of bids are uniformly distributed in the range of [10,50]. The default value of $t_{\max}$ is set to 60 [6].

Benchmark Algorithms. To evaluate the performance of A_{FL}, we provide a thorough analysis by comparing A_{FL} with the following benchmark schemes: compare it with three benchmark algorithms:

- *FedAvg* [4]: FedAvg selects clients randomly and averagely aggregates the weights of the local models from all selected clients.
- *Greedy Approximation Algorithm (GAA)* [22]: GAA greedily selects bid with the larger normalized value, which refers to the average bid's value per unit requested resource. The bid's value is concerned with the satisfaction level of the server and the bid's claimed cost. Here we adapt it by redefining the normalized value as $\left(\frac{T_l(\theta_{ij})}{\theta_{ij}} - b_{ij}\right)/(\eta_1 t_i^{com} + \eta_2 t_i^{cmp})$.
- A_{online} [19]: A_{online} first calculates the unit payment of each global iteration based on a payment function. Then it selects the client with larger utility and schedules the client according to the best schedule that maximizes its utility.

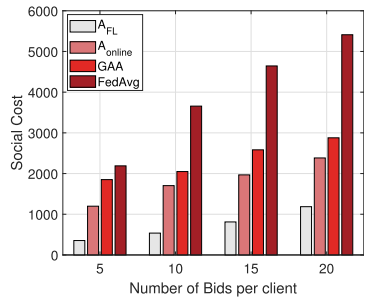

Fig. 6. Social cost under different J.

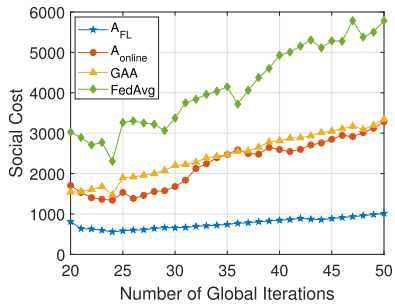

Fig. 7. Social cost at different fixed $\hat{T}_g$.

8.2 Performance of A_{FL}

Performance ratio. The *performance ratio* of an algorithm A for a minimization problem = the objective value of the solution found by A / the objective value returned by an optimal algorithm. We first study the performance ratio of A_{winner}. To ensure there are enough bids, we assume that all bids can satisfy constraint (5) and (7). Fig. 3 depicts the trend of A_{winner}'s performance ratio under different number of global iterations ($\hat{T}_g$) and bids per client (J). We can observe that A_{winner} has a small ratio (< 1.3) and the ratio becomes smaller as $\hat{T}_g$ decreases and J increases. This result is coincident with the theoretical analysis in Lemma 5 that $\hat{T}_g$ determines $H_{\hat{T}_g}$. In addition, the increase of J will decrease the length of time period (*i.e.* $|d_{ij} - a_{ij}|$) since we select $2J$ non-repeated random numbers to form J time periods for each client. The value of $\psi^t_{\min}$ increases when the length of time period decreases. Therefore, parameter ω eventually decreases. Next, we also study the impact of the number of clients (I) and bids per client (J) on performance ratio of A_{FL}. Fig. 4 shows performance ratios of all algorithms under different I and J. We can observe that the performance ratio of A_{FL} is the smallest and not affected greatly by the change of I and J. One reason is that A_{FL} can find the best solution by enumerating the number of global iterations T_g from T_0 to T.

Social Cost. Figs. 5 and 6 further plot the social cost under different numbers of clients (I) and bids per client (J). In both Figs. 5 and 6, we can see that A_{FL} outperforms three benchmark algorithms. Furthermore, the social cost of A_{FL} in Fig. 5 will decrease slightly with the increase of I since there is higher probability to select bids with lower average cost. On the contrary, the cost of all algorithms increase when the value of J increases in Fig. 6. Since the length of time periods will decrease if the number of bids per clients (J) increases, the

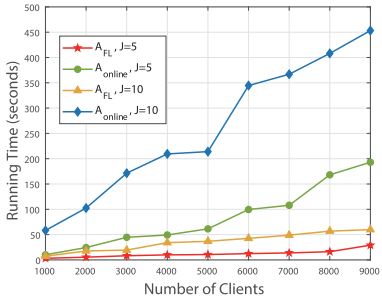

Fig. 8. Running time of A_{FL} and A_{online}.

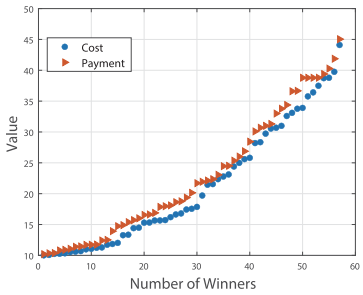

Fig. 9. Payment versus claimed cost of winning bid.

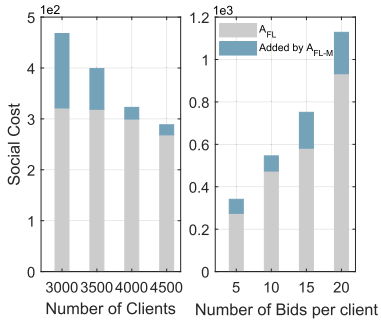

Fig. 10. Social cost under different I and J.

average cost gets higher when the claimed cost remains the same. As a result, the total cost of all algorithms become larger. Fig. 7 illustrates the social cost at different fixed $\hat{T}_g$ within the range $[T_0, T]$. From Fig. 7, we can find that A_{FL} still generates the lowest social cost. Moreover, we can see that all algorithms achieve the smallest cost when $\hat{T}_g = 26$. This is because the computation cost occupies a large proportion of the total cost at the early stage and it drops with the increase of $\hat{T}_g$. When the number of global iterations $\hat{T}_g$ is close to 26, all algorithms find the balance point between the computation and communication cost. After that, the total cost grows gradually with the increase of $\hat{T}_g$, since the communication cost dominates the total cost.

Running Time. In Fig. 8, we investigate the running time of A_{FL} and A_{online} under different number of clients, measured by *tic* and *toc* function in MATLAB. We evaluate the running time on our laptop with an Intel Core i7-4270HQ and 8-GB RAM memory. To minimize the error, we use the average result of five tests. We observe that the running time of A_{FL} is not affected greatly by the number of clients. Furthermore, A_{FL} can finish within 60 seconds even with a large input scale ($I = 9000$, $J = 10$), and runs fast than A_{online}.

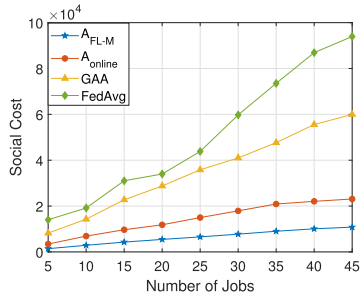

Fig. 11. Social cost under different numbers of jobs.

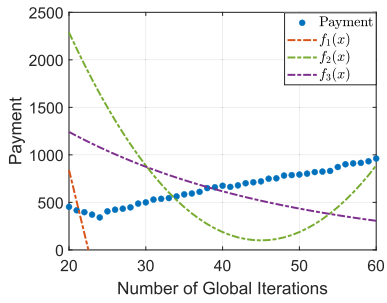

Fig. 12. Different budget functions.

Individual Rationality. Finally, Fig. 9 compares payment and claimed cost of all winners selected by A_{FL}. We can see that the payment for the winner is always larger than its corresponding claimed cost. Therefore, one can observe that the property of individual rationality can be satisfied and each winner's utility is non-negative.

8.3 Performance of of A_{FL-M}

Assume that there are 10 ($M = 10$) FL jobs that needed to train. We use a tailored exponential function $F_{Bm}(T_{gm}) = a * e^{-bT_{gm}}$ as the default budget function, where $a \in [1000, 2000], b \in [0.001, 0.01]$. Other parameter settings are consistent with the above setting. K_m are set within the range [10,20]. Similarly, we evaluate the performance of A_{FL-M} through large-scale simulations. Other benchmark algorithms are adjusted to accommodate to the scenario with multiple FL jobs. For ease of comparison, Fig. 10 depicts the social cost of A_{FL} and A_{FL-M} under different number of clients/bids per client. Especially, to demonstrate the interference of multiple jobs, we compare the average cost of one job achieved by A_{FL-M} with the cost of A_{FL}. We can observe that the difference between the cost of A_{FL} and A_{FL-M} decreases with the increase of number of clients. This is because there are more clients for the cloud platform to select. On the contrary, the difference increases with the increases of the number of bids per client. This phenomenon is reasonable since the large number of bids per client means that client preemption will be more likely to happen, whereas at most one bid of each client can be accepted according to constraint (17 f). Fig. 11 investigates the performance of algorithms under different numbers of jobs (M). The result from Fig. 11 illustrates that the social cost of A_{FL-M} increases with the number of jobs. Especially, we can observe that social cost of A_{FL-M} does not increase any more when it meets a larger number of jobs because of resource scarcity (client).

TABLE 2
Parameter Settings of Three Datasets

Dataset	First/Second layer output channels	# of Batch sizes	T
MNIST [51]	20/50	10	50
Fashion-MNIST [52]	16/32	100	50
CIFAR-10 [53]	6/16	50	70

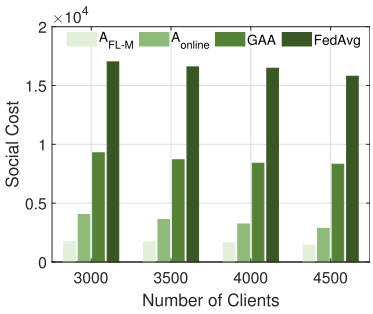

Fig. 13. Social cost under different I.

To test the effect of the sensitivity of the FL job's budget function, we adopt three types of typical functions as follows [50].

- Linear Function: $f_1(x) = -ax + b$, - $a \in [10, 20], b \in [1000, 2000]$.
- Polynomial Function: $f_2(x) = ax^2 + bx + c$, - $a \in [0.01, 0.1], b \in [10, 15], c \in [1000, 2000]$.
- Exponential Function (default): $f_3(x) = a * e^{-bx}$, - $a \in [1000, 2000], b \in [0.001, 0.01]$.

We conduct five independent tests of A_{FL-M} and record the average results in Fig. 12. To satisfy constraint (17 g), the value of budget function must be at least the corresponding payment. In other words, the region that the curve of budget function above the payment is available for A_{FL-M}. In Fig. 12, we can observe that functions $f_2(x)$ and $f_3(x)$ can achieve better performance than function $f_1(x)$, *i.e.*, the minimum payments of $f_2(x)$ and $f_3(x)$ are both smaller than $f_1(x)$'s. Therefore, the balance between communication and computation (*i.e.*, the minimum social cost) may be not achieved under the interference of FL jobs' budget functions.

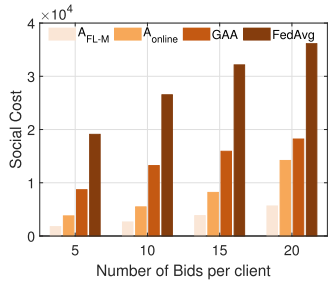

Fig. 14. Social cost under different J.

9 Testbed Implementation

In this section, we conduct testbed experiments by adopting FL frameworks FAVOR [23] and CoCoA [8] to evaluate the performance of our algorithm. Here, we only present the results of implementing A_{FL-M} since A_{FL} and A_{FL-M} both not affect the training process.

9.1 Implementation

Note that our proposed auction methods focus more on the auction process and are actually client selection algorithms. Therefore, the real implementation of the auction process can be taken as a selection component to combine within existing federated platforms. The information exchanging steps that involved in the auction process can be simply achieved through the FL platform's inherent Internet connection between the FL platform and clients. With the Python threading library, we adopt and simulate numerous clients in the FL environment by using lightweight threads. Each thread (*i.e.*, client) will run one real-world PyTorch model on three typical datasets MNIST, Fashion-MNIST and CIFAR-10. In the auction process, all clients will submit their bids to one virtual machine which served as the cloud server. Limited by the computation scale, we only simulate the winners as threads rather than all clients. Besides, the training data of each client is a subset of the typical dataset. Unless explicitly stated, parameters (*e.g.*, number of local iterations $T_l(\theta_{ij})$) are consistent with the settings of simulation. To guarantee the theoretical convergence of the FL model, we adopt the parameter aggregation methods of CoCoA [8] corresponding to the training accuracy to update the global model. In our experiments, we train a CNN model with two 5×5 convolution layers, with each layer

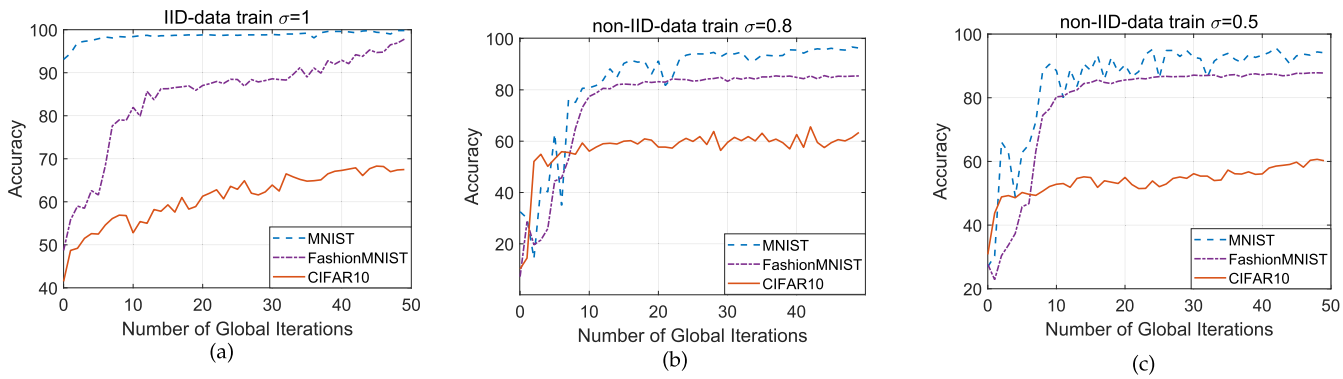

Fig. 15. FL training process under different levels of non-IID data.

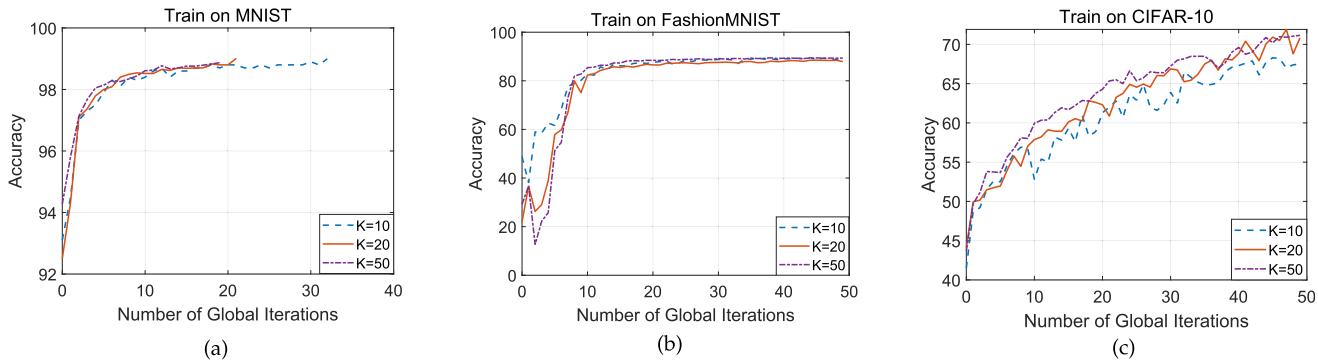

Fig. 16. FL training process under different levels of parallelism.

followed by 2×2 max pooling. Note that the number of clients per global iteration (K) is set to 10. In addition, several specific parameters of the datasets are listed in Table 2.

9.2 Evaluation Results

Social Cost. Figs. 13 and 14 demonstrate that the social cost under different number of clients/bids per client, respectively. Compared with the simulation results shown in Section 8, the results is similar. That can be a strong evidence to state that our proposed algorithm always outperforms *FedAvg* [4], *GAA* [22] and A_{online} [19] even in testbed experiments.

Different levels of Non-IID Data. Recall that we assume all clients' data follow an identical and independent distribution, *i.e.*, IID. However, in practice, the data distribution among data samples of clients is usually not independent and identically distributed (IID), which is also a main concern in FL. Therefore, we further investigate the accuracy of FL training process under different levels of non-IID data. For clarity, we use one specific metric σ to represent different levels of non-IID data. The metric $\sigma \in [0, 1]$ denotes the proportion of data of other labels that including in the dataset of one client. In this regard, the smaller σ is, the stronger the non-IID distribution of data samples among clients is. For instance, $\sigma = 0.6$ indicating that 60% of the data belongs to other labels and the remaining 40% of the data belongs to one corresponding label. Here, Fig. 15 depicts the accuracy tendency on three different levels of non-IID data. Our experiment results show that it needs more number of global iterations when σ decreases.

Different levels of Parallelism. In addition, we study the effect of parallelism (*i.e.*, the number of clients per global iteration, K) in the FL training process, shown in Fig. 16. Figs. 16a, 16b and 16c further demonstrate the accuracy tendency of the FL job's training process on MNIST, Fashion-MNIST and CIFAR-10, respectively. We can observe that the performance of federated learning does not be affected greatly when increasing the parallelism, *i.e.*, the value of K.

10 Conclusion

Federated learning (FL) is shown as a remarkable privacy-preserving approach to train machine learning jobs without exchanging data samples. Besides technical challenges that are being studied in the literature, economic incentives of such distributed machine learning process is also critical for realizing practical applications. In this paper, we propose a reverse auction to incentivize the participation of heterogeneous clients. Different from previous research, we select and schedule winners (or mobile clients) to execute training job in different global iterations, with a goal of social cost minimization. In addition, the number of global iterations is determined by the global accuracy and local accuracy. Both theoretical analysis and large-scale simulations based on the real-world data verified that our proposed auction is truthful, individual rational, computationally efficient, and achieves near-optimal social cost. The superiority of our algorithm over benchmark algorithms is also confirmed by large-scale simulations and testbed experiments. In addition, we discuss one realistic scenario where there are multiple FL jobs with corresponding budget functions and further propose an effective solution.

In practice, one may not be able to obtain the actual local accuracy and there may be some variations in the training process due to hardware specifications. Furthermore, clients may drop out with high probability since the network connection (4G or WiFi) can be unstable. As a future direction, it is interesting to further study a more realistic scenario that combines these considerations.

References

[1] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, 2015, pp. 1310–1321.
[2] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *Proc. IEEE Symp. Secur. Privacy*, 2017, pp. 19–38.
[3] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*.
[4] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Statist.*, 2017, pp. 1273–1282.
[5] Gboard, now available for Android, 2016. [Online]. Available: https://blog.google/products/search/gboard-now-on-android/
[6] N. H. Tran, W. Bao, A. Zomaya, N. M. NH, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE Conf. Comput. Commun.*, 2019, pp. 1387–1395.
[7] M. Jaggi *et al.*, "Communication-efficient distributed dual coordinate ascent," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 3068–3076.
[8] C. Ma *et al.*, "Distributed optimization with arbitrary local solvers," *Optim. Methods Softw.*, vol. 32, no. 4, pp. 813–848, 2017.
[9] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, *arXiv: 1812.06127*.

[10] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 4427–4437.
[11] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. Commun.*, 2019, pp. 1–7.
[12] S. Wang *et al.*, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.
[13] K. Toyoda and A. N. Zhang, "Mechanism design for an incentive-aware blockchain-enabled federated learning platform," in *Proc. IEEEInt. Conf. Big Data*, 2019, pp. 395–403.
[14] S. R. Pandey, N. H. Tran, M. Bennis, Y. K. Tun, A. Manzoor, and C. S. Hong, "A crowdsourcing framework for on-device federated learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3241–3256, May 2020.
[15] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10 700–10 714, Dec. 2019.
[16] R. Zeng, S. Zhang, J. Wang, and X. Chu, "FMore: An incentive scheme of multi-dimensional auction for federated learning in MEC," *Proc. 40th Int. Conf. Distrib. Comput. Syst.*, 2020, pp. 278–288.
[17] W. Y. B. Lim *et al.*, "Hierarchical incentive mechanism design for federated machine learning in mobile networks," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9575–9588, Oct. 2020.
[18] M. Dai, Z. Su, Y. Wang, and Q. Xu, "Contract theory based incentive scheme for mobile crowd sensing networks," in *Proc. IEEE Int. Conf. Sel. Top. Mobile Wireless Netw.*, 2018, pp. 1–5.
[19] R. Zhou, Z. Li, C. Wu, and Z. Huang, "An efficient cloud market mechanism for computing jobs with soft deadlines," *IEEE/ACM Trans. Netw.*, vol. 25, no. 2, pp. 793–805, Apr. 2017.
[20] R. B. Myerson, "Optimal auction design," *Math. Operations Res.*, vol. 6, no. 1, pp. 58–73, 1981.
[21] A. Archer and É. Tardos, "Truthful mechanisms for one-parameter agents," in *Proc. IEEE Symp. Found. Comput. Sci.*, 2001, pp. 482–491.
[22] T. H. T. Le *et al.*, "An incentive mechanism for federated learning in wireless cellular network: An auction approach," *IEEE Trans. Wireless Commun.*, vol. 20, no. 8, pp. 4874–4887, Aug. 2021.
[23] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-IID data with reinforcement learning," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 1698–1707.
[24] S. Wang, M. Lee, S. Hosseinalipour, R. Morabito, M. Chiang, and C. G. Brinton, "Device sampling for heterogeneous federated learning: Theory, algorithms, and implementation," in *Proc. IEEE Int. Conf. Commun.*, 2021, pp. 1–10.
[25] Y. Zhan, P. Li, and S. Guo, "Experience-driven computational resource allocation of federated learning by deep reinforcement learning," in *Proc. Int. Parallel Distrib. Process. Symp.*, 2020, pp. 234–243.
[26] B. Luo, X. Li, S. Wang, J. Huang, and L. Tassiulas, "Cost-effective federated learning design," in *Proc. IEEE Int. Conf. Commun.*, 2021, pp. 1–10.
[27] Y. Deng *et al.*, "FAIR: Quality-aware federated learning with precise user incentive and model aggregation," in *Proc. IEEE Int. Conf. Commun.*, 2021, pp. 1–10.
[28] Z. Wang, H. Xu, J. Liu, H. Huang, C. Qiao, and Y. Zhao, "Resource-efficient federated learning with hierarchical aggregation in edge computing," in *Proc. IEEE Int. Conf. Commun.*, 2021, pp. 1–10.
[29] A. Li *et al.*, "Sample-level data selection for federated learning," in *Proc. IEEE Int. Conf. Commun.*, 2021, pp. 1–10.
[30] K. Wei *et al.*, "User-level privacy-preserving federated learning: Analysis and performance optimization," *IEEE Trans. Mobile Comput.*, early access, Feb. 4, 2021, doi: 10.1109/TMC.2021.3056991.
[31] X. Zhang, Z. Yang, Y. Liu, J. Li, and Z. Ming, "Toward efficient mechanisms for mobile crowdsensing," *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 1760–1771, Feb. 2017.
[32] R. Zhou, Z. Li, and C. Wu, "A truthful online mechanism for location-aware tasks in mobile crowd sensing," *IEEE Trans. Mobile Comput.*, vol. 17, no. 8, pp. 1737–1749, Aug. 2018.
[33] L. Duan, T. Kubo, K. Sugiyama, J. Huang, T. Hasegawa, and J. Walrand, "Motivating smartphone collaboration in data acquisition and distributed computing," *IEEE Trans. Mobile Comput.*, vol. 13, no. 10, pp. 2320–2333, Oct. 2014.
[34] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing," in *Proc. ACM 18th Annu. Int. Conf. Mobile Comput. Netw.*, 2012, pp. 173–184.
[35] N. Ding, Z. Fang, and J. Huang, "Optimal contract design for efficient federated learning with multi-dimensional private information," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 186–200, Jan. 2021.
[36] J. Weng, J. Weng, H. Huang, C. Cai, and C. Wang, "Fedserving: A federated prediction serving framework based on incentive mechanism," in *Proc. IEEE Int. Conf. Commun.*, 2021, pp. 1–10.
[37] T. H. T. Le, N. H. Tran, Y. K. Tun, Z. Han, and C. S. Hong, "Auction based incentive design for efficient federated learning in cellular wireless networks," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2020, pp. 1–6.
[38] X. Lin, J. Wu, J. Li, X. Zheng, and G. Li, "Friend-as-learner: Socially-driven trustworthy and efficient wireless federated edge learning," *IEEE Trans. Mobile Comput.*, early access, Apr. 21, 2021, doi: 10.1109/TMC.2021.3074816.
[39] M. Tang and V. W. Wong, "An incentive mechanism for cross-silo federated learning: A public goods perspective," in *Proc. IEEE Int. Conf. Commun.*, 2021, pp. 1–10.
[40] Y. Jiao, P. Wang, D. Niyato, B. Lin, and D. I. Kim, "Toward an automated auction framework for wireless federated learning services market," *IEEE Trans. Mobile Comput.*, vol. 20, no. 10, pp. 3034–3048, Oct. 2021.
[41] W. Y. B. Lim *et al.*, "Decentralized edge intelligence: A dynamic resource allocation framework for hierarchical federated learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 3, pp. 536–550, Mar. 2021.
[42] J. S. Ng *et al.*, "A hierarchical incentive design toward motivating participation in coded federated learning," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 359–375, Jan. 2022.
[43] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.
[44] J. Kleinberg and E. Tardos, *Algorithm Design*. Noida, Uttar Pradesh, India: Pearson Education India, 2006.
[45] D. G. Luenberger, *Introduction to Linear and Nonlinear Programming*. Reading, MA, USA: Addison-Wesley, 1973, vol. 28.
[46] M. Elloumi, R. Dhaou, B. Escrig, H. Idoudi, and L. A. Saidane, "Monitoring road traffic with a UAV-based system," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2018, pp. 1–6.
[47] X. Jiang *et al.*, "FGLP: A federated fine-grained location prediction system for mobile users," 2021, *arXiv:2106.08946*.
[48] P. Chhikara, R. Tekchandani, N. Kumar, M. Guizani, and M. M. Hassan, "Federated learning and autonomous uavs for hazardous zone detection and aqi prediction in iot environment," *IEEE Internet Things J.*, vol. 8, no. 20, pp. 15456–15467, Oct. 2021.
[49] F. Miao, L. Yao, and X. Zhao, "Symbiotic organisms search algorithm using random walk and adaptive cauchy mutation on the feature selection of sleep staging," *Expert Syst. Appl.*, vol. 176, 2021, Art. no. 114887.
[50] Z. Chi *et al.*, "Online dispatching and scheduling of jobs with heterogeneous utilities in edge computing," in *Proc. ACM 21st Int. Symp. Theory Algorithmic Found. Protoc. Des. Mobile Netw. Mobile Comput.*, 2020, pp. 101–110.
[51] The MNIST Dataset, 1998. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[52] The Fashion-MNIST dataset, 2017. [Online]. Available: https://github.com/zalandoresearch/fashion-mnist
[53] The CIFAR-10 Dataset, 2009. [Online]. Available: https://www.cs.toronto.edu/kriz/cifar.html

Jinlong Pang received the BE degree from the School of Power and Machinery and the second BE degree from the School of Computer both from Wuhan University, China. He is currently working toward the ME degree with the School of Cyber Science and Engineering, Wuhan University. His research interests include distributed machine learning, federated learning, online learning, and algorithm optimization.

Ruiting Zhou (Member, IEEE) received the PhD degree from the Department of Computer Science, University of Calgary, Canada, in 2018. She has been an associate professor with the School of Cyber Science and Engineering, Wuhan University since June 2018. Her research interests include cloud computing, machine learning and mobile network optimization. She has published research papers in top-tier computer science conferences and journals, including IEEE INFOCOM, ACM MobiHoc, ICDCS, *IEEE/ACM Transactions on Networking*, *IEEE Journal on Selected Areas in Communications*, and *IEEE Transactions on Mobile Computing*. She also serves as a reviewer for journals and international conferences such us the *IEEE Journal on Selected Areas in Communications*, *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Cloud Computing*, *IEEE Transactions on Wireless Communications*, and IEEE/ACM IWQOS.

Jieling Yu received the BE degree from the School of Cyber Science and Engineering, Wuhan University, China, in 2021. She is currently working toward the master's degree from the School of Cyber Science and Engineering, Wuhan University, China. Her research interests include edge computing, federated learning, online learning, and network optimization.

John C.S. Lui (Fellow, IEEE) received the PhD degree in computer science from UCLA. He is currently the Choh-Ming Li chair professor with the Department of Computer Science & Engineering (CSE), The Chinese University of Hong Kong (CUHK). After his graduation, he joined the IBM Laboratory and participated in research and development projects on file systems and parallel I/O architectures. He later joined the CSE Department, CUHK. His current research interests include in online learning algorithms and applications (e.g., multi-armed bandits, reinforcement learning), machine learning on network sciences and networking systems, large scale data analytics, network/system security, network economics, large scale storage systems and performance evaluation theory. He received various departmental teaching awards and the CUHK Vice-Chancellor's Exemplary Teaching Award. John also received the CUHK Faculty of Engineering Research Excellence Award (2011-2012). He is an elected member of the IFIP WG 7.3, fellow of ACM, senior research fellow of the Croucher Foundation, fellow of the Hong Kong Academy of Engineering Sciences (HKAES), and was the past chair of the ACM SIGMETRICS (2011-2015). His personal interests include films and general reading.