

# Fair Concurrent Training of Multiple Models in Federated Learning

Marie Siew\*, Haoran Zhang†, Jong-Ik Park†, Yuezhou Liu‡, Yichen Ruan, Lili Su‡, Stratis Ioannidis‡,  
Edmund Yeh‡, and Carlee Joe-Wong†

\*Information Systems Technology and Design Pillar, Singapore University of Technology and Design, 487372 Singapore

†Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA

‡Electrical and Computer Engineering, Northeastern University, Boston, MA 02115 USA

Emails: marie\_siew@sutd.edu.sg, {haoranz5, jongikp, cjoewong}@andrew.cmu.edu, {liu.yuez, l.su}@northeastern.edu, yichenr@alumni.cmu.edu, stratis.ioannidis@gmail.com, eyeh@ece.neu.edu

**Abstract**—Federated learning (FL) enables collaborative learning across multiple clients. In most FL work, all clients train a single learning task. However, the recent proliferation of FL applications may increasingly require multiple FL tasks to be trained simultaneously, sharing clients’ computing and communication resources, which we call Multiple-Model Federated Learning (MMFL). Current MMFL algorithms use naïve average-based client-task allocation schemes that can lead to unfair performance when FL tasks have heterogeneous difficulty levels, e.g., tasks with larger models may need more rounds and data to train. Just as naïvely allocating resources to generic computing jobs with heterogeneous resource needs can lead to unfair outcomes, naïve allocation of clients to FL tasks can lead to unfairness, with some tasks having excessively long training times, or lower converged accuracies. Furthermore, in the FL setting, since clients are typically not paid for their training effort, we face a further challenge that some clients may not even be willing to train some tasks, e.g., due to high computational costs, which may exacerbate unfairness in training outcomes across tasks. We address both challenges by firstly designing FedFairMMFL, a difficulty-aware algorithm that dynamically allocates clients to tasks in each training round. We provide guarantees on fairness and FedFairMMFL’s convergence rate. We then propose a novel auction design that incentivizes clients to train multiple tasks, so as to fairly distribute clients’ training efforts across the tasks. We show how our fairness-based learning and incentive mechanisms impact training convergence and finally evaluate our algorithm with multiple sets of learning tasks on real world datasets.

**Index Terms**—Federated Learning, Fair Resource Allocation, Incentive Mechanism

## I. INTRODUCTION

Federated learning (FL) enables multiple devices or users (called clients) to collectively train a machine learning model [1]–[4], with data staying at the user side. Several recent works have demonstrated FL’s potential in many applications, such as smart homes [2], [3], analyzing healthcare data [5], and next word prediction [6]. Despite the large and growing set of FL applications, however, the vast majority of FL research assumes that each client only trains one learning task during FL. In practice, clients may concurrently train multiple tasks, e.g., if smartphones contribute to training both next-word prediction and advertisement recommendation models, or if hospitals contribute to training separate prediction models for heart disease and breast cancer. Clients could also train

multiple versions of a model, in order to select the best-performing one for deployment, e.g., training many different neural network architectures on the same data. We call this setting in which FL clients simultaneously train multiple tasks, *Multiple-Model Federated Learning* (MMFL) [7]–[11]. Training multiple tasks in one FL setting amortizes the real-world costs of recruiting clients and coordinating the training time across clients, improving resource efficiency.

Since FL clients, especially mobile or Internet of Things devices, may be resource-constrained [1], they may not be able to train every FL task at the same time. Thus, we must allocate clients to the different tasks in each training round. Prior work on MMFL has shown that with careful selection of clients for each task in each training round, concurrent training yields comparable accuracy to training tasks separately [7]. These works, however, seek to optimize the *average* task accuracy or training time, and often assume that training tasks have *similar difficulties*, e.g., training multiple copies of the same model [7], [8]. In practice, however, FL tasks may be requested by different entities and users, and it is important to ensure comparable convergence time and converged accuracies across tasks of *varied difficulty levels*. Hospitals, for example, may wish to concurrently train prediction models for different diseases, without favoring one disease’s model over another. Likewise, in applications like autonomous driving, improving the worst-performing FL task helps to improve the overall quality of experience for users. Indeed, even if the multiple tasks are different models trained by the same FL operator, the operator will not know which model performs best until the training of *all* models is complete: we therefore would want to dynamically allocate clients to tasks so as to optimize the performance of the worst-performing task, which is accomplished by optimizing for fairness. To the best of our knowledge, we are the first to extend the MMFL setting to consider **fair training** of concurrent FL models.

### A. Research Challenges in Fair MMFL

In considering fairness across multiple different models, we seek to ensure that each model achieves **comparable training performance**, in terms of their converged accuracies, or the time taken to reach threshold accuracy levels. The

first challenge to doing so is that **the multiple models can have varying difficulties** and thus require different amounts of data and computing resources to achieve the same accuracy. Existing MMFL training algorithms, e.g., the round-robin algorithm proposed in [7], allocate clients to tasks uniformly, or assume that tasks have similar or the same difficulty, e.g., training multiple copies of the same model [8]. However, while such strategies arguably achieve fairness, or at least equality, in resource (i.e., client) allocation to different models, they may lead to “unfair” outcomes. For example, smart home devices may train simple temperature prediction models that take low-dimensional timeseries as input data, as well as complex person identification ones that use high-dimensional image input data [3]. Allocating equal resources to each model may then yield poor performance for the more difficult ones.

Much work in computer systems and networks has focused on fairness in allocating resources to different computing jobs or network flows, given their heterogeneous resource requirements [12]–[14]. Simply applying these allocation algorithms to MMFL, however, is difficult: here, each “resource” is a client, and the performance improvement realized by allocating this client to a given model’s training task is unknown in advance and will change over time. This is because it depends on the current model state and the data distribution across clients, both of which are highly stochastic. We must therefore handle significant **uncertainty and dynamics** in how much each client contributes to each task’s training performance.

Finally, purely viewing MMFL fairness as a matter of assigning clients to tasks ignores the fact that **clients may not be equally willing to contribute resources to each model**. For example, smartphone users may benefit from better next-word prediction models if they text frequently, but may derive limited benefit from a better ad recommendation model. Users may refuse to compute updates if they judge that associated computing costs outweigh personal benefits. If multiple users prefer one model over another, then the disfavored model will overfit to users willing to train it, again leading to an unfair outcome. In an extreme case, no users may be assigned to train this task at all. Prior work [15] has proposed mechanisms to incentivize clients to help train multiple non-simultaneously trained FL models, but the focus was on optimizing the task take-up rate, instead of ensuring that clients are *fairly distributed* across models.

### B. Our Contributions

Our goal is to design methods to ensure fair concurrent training of MMFL, addressing the challenges above. To do so, we utilize the popular metric of  $\alpha$ -fairness, commonly used for fair resource allocation in networking [16] and previously proposed to measure fairness across clients in training a single FL model [17]. We use  $\alpha$ -fairness and Max-Min fairness (a special case of  $\alpha$ -fairness when  $\alpha \rightarrow \infty$  [16]) to design novel algorithms that both *incentivize* clients to contribute to FL tasks and *adaptively allocate* these incentivized clients to multiple tasks during training. Intuitively, these algorithms ensure that client effort is spread across the MMFL tasks in

accordance with their difficulty, helping to achieve fairness with respect to task performances. After discussing related works in Section II, we make the following contributions:

- We first suppose that all clients are willing to train all models, and propose **FedFairMMFL**, an **algorithm to fairly allocate clients to tasks** (i.e., models) in Section III. Our algorithm dynamically allocates clients to tasks according to the current performance levels of all tasks. We design **FedFairMMFL** to approximately optimize the  $\alpha$ -fairness of all tasks’ training accuracies.
- From the perspective of a single model, our client-task allocation algorithm in MMFL can be viewed as a form of *client selection*. We therefore adapt and refine convergence guarantees from analyses of client selection in FL to **analyze the convergence of MMFL tasks under FedFairMMFL** in Section IV, showing that **FedFairMMFL** preferentially accelerates convergence of more difficult tasks (Theorem 4 and Corollary 5) and allows each model’s training to converge (Corollary 6).
- We next suppose that clients may have unequal interest in training different tasks and may need to be incentivized. We discuss *Budget-Fairness* and *Max-min Fairness*, and we propose a **max-min fair auction mechanism** to ensure that a fair number of clients are incentivized across tasks. We show that it incentivizes truthful client bids with high probability (Theorem 8 and Corollary 9). We also show that it leads to more fair outcomes: it allocates more clients to each model, compared to other auction mechanisms that either ignore fairness or allocate the same budget to each task (Corollary 10). Following which, we provide a convergence bound showing how our proposed incentive mechanisms coupled with **FedFairMMFL** impact the convergence rate (Theorem 11).
- We finally **validate** our client incentivization and allocation algorithms on multiple FL datasets. Our results show that our algorithm achieves a higher minimum accuracy across tasks than random allocation and round robin baselines, and a lower variance across task accuracy levels, while maintaining the average task accuracy. Our max-min client incentivization further improves the minimum accuracy in the budget-constrained regime.

We discuss future directions and conclude in Section VII.

## II. RELATED WORK

We distinguish our MMFL setting from that of **multi-task learning (MTL)**, in which multiple training tasks have the same or a similar structure [18], e.g., a similar underlying lower rank structure. Unlike clustered FL [19], [20], our work is different from combining MTL with FL, instead we consider multiple *distinct* learning tasks. Our analysis of MMFL performance is closer to prior convergence analysis of **client selection in FL** [21]–[23]. We adapt prior work on FL convergence to reveal the relationships between model convergence and our definition of fairness across tasks.

Past work on **multi-model federated learning** also aims to optimize the allocation of clients to model training tasks.

To the best of our knowledge, we are the first to explicitly consider fairness across the different tasks. Some algorithms focus on models with similar difficulties, for which fairness is easier to ensure [7], while other works propose reinforcement learning [24], heuristic [9], or asynchronous client selection algorithms [10] to optimize average model performance. Other work optimizes the training hyperparameters [25] and allocation of communication channels to tasks [21].

**Fair allocation of multiple resources** has been considered extensively in computer systems and networks [12], [13], [26]. In crowdsourcing frameworks, the fair matching of clients to tasks has been studied in energy sensing [27], delivery [28], and social networks [29]. However, unlike MMFL tasks, individual crowdsourcing tasks generally do not need multiple clients. Fair resource allocation across tasks in FL differs from all the above applications as the amount of performance improvement realized by allocating one (or more) clients to a given model's training task is unknown and stochastic, dependent on the model state and clients' data distributions. In FL, fairness has generally been considered from the *client* perspective, e.g., accuracies or contribution when training a single model [17], [30], and not across distinct training tasks.

Many **incentive mechanisms** have been proposed for FL and other crowdsourcing systems [31]–[35], including auctions for clients to participate in many FL tasks [15]. To the best of our knowledge, this is the first time client incentives have aimed to ensure fair training of multiple tasks in MMFL.

### III. FAIR CLIENT TASK ALLOCATION FOR MULTIPLE-MODEL FL

We aim to train multiple models (i.e., tasks)<sup>1</sup> concurrently, such that there is fair *fair* training performance across models, e.g., in terms of the time taken, or converged accuracies. Models may be trained on distinct or the same datasets. This performance is a function of which clients train which model at which training round. In this section, we assume clients always train any given model upon request and propose a client-model (i.e., client-task) allocation scheme that enables fair training of the multiple tasks. In Sec. V, we introduce auction mechanisms, which incentivize clients to commit to tasks at a fair rate across tasks, before training begins (Fig. 1).

**MMFL framework:** Consider  $K$  clients collaboratively training  $S$  distinct machine learning tasks in MMFL (Fig. 1). If  $S = 1$ , our setting reduces to traditional FL. Each client is equipped with local data for training each task. All clients can communicate with a central server that maintains a global model  $\mathbf{w}_s$  for each task  $s \in \mathbb{S}$ . Let each client  $k$ 's local loss over its local dataset  $\mathcal{D}_{k,s}$  for task  $s$  be  $F_{k,s}(\mathbf{w}_s) = \sum_{d \in \mathcal{D}_{k,s}} \ell(d, \mathbf{w}_s) / |\mathcal{D}_{k,s}|$ , where  $\ell$  is a loss function that measures the accuracy of the model  $\mathbf{w}_s$  on data point  $d$ . We then define the global loss for task  $s$  as a weighted sum over the local losses:  $f_s(\mathbf{w}_s) = \sum_{k=1}^K p_{k,s} F_{k,s}(\mathbf{w}_s)$ , where  $p_{k,s} = \frac{|\mathcal{D}_{k,s}|}{\sum_k |\mathcal{D}_{k,s}|}$  is the fraction of training data for task  $s$  that

<sup>1</sup>We use model and task interchangeably in this paper.

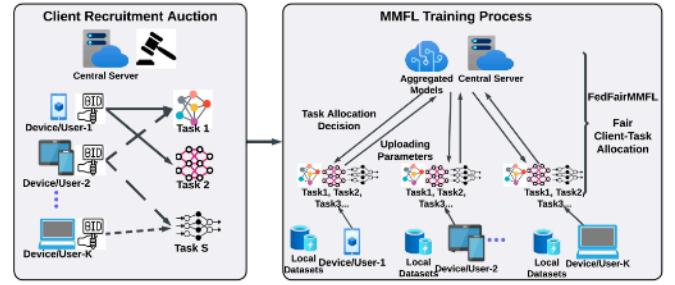


Fig. 1. Fair Multiple Model Federated Learning to ensure that each model achieves comparable training performance: The pipeline's first part involves *incentive-aware* client recruitment auctions to ensure a fair take-up rate over tasks. This is followed by a *difficulty-aware* client-task allocation scheme that allocates tasks to clients based on the current loss levels of all tasks.

resides at client  $k$ . Similar to traditional FL, in MMFL, the usual aim is to minimize the sum of losses [10]:

$$\min_{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_S} \sum_{s=1}^S f_s(\mathbf{w}_s), \quad (1)$$

**Fairness over tasks:** We introduce *fairness* into this objective by utilizing the celebrated  $\alpha$ -fair utility function often employed to ensure fair network resource allocation [13], [16]. Viewing the loss of each model as an inverse “utility,” we then define our  $\alpha$ -fair MMFL objective as

$$\min_{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_S} g^\alpha(\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_S) \equiv \sum_{l=1}^S f_l^\alpha(\mathbf{w}_l), \quad (2)$$

where  $f_s^\alpha$  is task  $s$ 's  $\alpha$ -fair global loss function. This objective function helps to improve the uniformity of the converged losses, as we will analytically show in Section IV. Here the parameter  $\alpha \in [1, \infty)$  controls the emphasis on fairness; for  $\alpha = 1$ , the objective is to minimize the average global loss across tasks. As  $\alpha$  grows, we put more emphasis on fairness,<sup>2</sup> because analytically, a higher  $\alpha$  will give higher priority to the terms in the objective with larger  $f_s$ , during minimization.

To solve (2) within the FL framework, we will use gradient descent to design a training algorithm given the gradient

$$\nabla_{\mathbf{w}_s} g = \alpha f_s^{\alpha-1}(\mathbf{w}_s) \sum_k p_k \nabla F_{k,s}(\mathbf{w}_s). \quad (3)$$

**FedFairMMFL training (Algorithm 1):** Our MMFL training begins with the initialization of the global model weights  $\mathbf{w}_s$  for all tasks, along with an initial random client-task allocation (line 1). The algorithm then enters a loop over global rounds (lines 2–15). During each global round  $t$ ,  $m = C \times |\mathcal{C}|$  clients are active (line 3). Here  $C$  is the rate at which clients are active. For all active clients, task allocation happens. We let  $Sel_s(t)$  denote the set of clients allocated to task  $s$  in round  $t$ , dropping the subscript  $s$  if it is clear from context. Following prior work on MMFL [7], [8], [24], we *allocate each client to at most one task during each training round*, i.e., the  $Sel_s(t)$  are disjoint across tasks  $s$ . For example, mobile clients or

<sup>2</sup>Usually,  $\alpha$ -fair functions aim to *maximize* a sum of utility functions  $(1 - \alpha)^{-1} \sum_s U_s^{1-\alpha}$ , where  $U_s$  is the utility of task  $s$ . We instead aim to *minimize* the sum of  $\alpha$ -fair loss functions, viewing the loss as an *inverse* utility.

other resource constrained devices may be performing other background tasks, e.g., running multiple applications, and thus may not have the computing power or battery to complete training rounds of multiple tasks.

Intuitively, uniform client-task allocation methods, like random or round robin allocation [8], will train simpler tasks faster, since they allocate an equal number of clients to each task, in expectation. We would instead like to ensure tasks are trained at more similar rates (i.e., achieve more fair performance) with a **difficulty-aware** client-task allocation algorithm, in which more users are allocated to tasks that currently have a higher loss compared to other tasks. We can do so by inspecting the gradient of our MMFL objective in (3). Since resource-constrained clients can only compute gradients for one task per training round, we cannot exactly compute these gradients. Instead, we will assign each client to contribute to the gradient  $\nabla F_{k,s}(\mathbf{w}_s)$  for each task  $s$  at a probability in proportion to its weight for that task, i.e., proportional to  $\alpha f_s(\mathbf{w}_s)^{\alpha-1} p_k$ . We therefore allocate a task to each client according to the probability vector (lines 4-5):

$$\mathbf{p} = \left[ \frac{f_1^{\alpha-1}}{\sum_{s \in \mathbb{S}} f_s^{\alpha-1}}, \frac{f_2^{\alpha-1}}{\sum_{s \in \mathbb{S}} f_s^{\alpha-1}}, \dots, \frac{f_S^{\alpha-1}}{\sum_{s \in \mathbb{S}} f_s^{\alpha-1}} \right], \quad (4)$$

This allocation fulfills our intuition: **tasks currently with higher losses  $f_s$  have a higher probability of being allocated clients.** For example, when  $\alpha \rightarrow \infty$ , all clients will be allocated to the worst performing task, and when  $\alpha = 1$ , allocation will be uniform. In Section IV, we formalize this intuition by deriving a convergence bound for each task using **FedFairMMFL**. This selection scheme is *unbiased* across clients: each client is equally likely to train a given task.

Given the clients allocated to each task  $s$  at time  $t$ ,  $Sel_s(t)$ , the server then sends the latest global model weights  $\mathbf{w}_s(t)$  to task  $s$ 's allocated clients (lines 7-8). Each client then trains its task on its corresponding local dataset by running  $\tau$  SGD steps on the local loss  $F_{k,s}$  of its assigned task  $s$ , as in traditional FL [1] (line 9). Let  $\mathbf{w}_{k,s}(t)$  denote the updated local model for active client  $k$  on its allocated task  $s$ . Each active client then uploads  $\mathbf{w}_{k,s}(t)$  to the server (line 9), which aggregates the model parameters to synchronize the global model. This aggregation uses weights  $p_{k,Sel} = p_k / \sum_{k' \in Sel} p_{k'}$  for each client  $k$  (line 12), where  $Sel$  is the set of clients training task  $s$  in this training round. The server then obtains the global loss for each task  $f_s$ , to guide allocation in the next training round (line 13). The next training round then begins.

#### IV. FAIRNESS AND CONVERGENCE GUARANTEES

We now analyze **FedFairMMFL**'s performance. The full versions of all our proofs are in our Technical Report [36].

##### A. Fairness Guarantees

We first analyze the optimal solution to our fairness objective (2) to show that it reduces the variation in training

---

##### Algorithm 1 FedFairMMFL: Fair Training of Multiple Models in Federated Learning

---

```

1: Initialization: The global model weights  $\mathbf{w}_s$ , for all tasks, initial
   random client-task allocation.
2: for  $Globalepoch = 1, 2, \dots$  do
3:    $m = C \times |\mathcal{C}|$  clients are active.
4:   Server determines task allocation for active clients,
5:   based on  $\mathbf{p} = [f_1^{\alpha-1} / \sum_{s \in \mathbb{S}} f_s^{\alpha-1}, \dots]$  (Eq. 4).
6:   for all active clients  $k$  do
7:     Server sends latest global model weights of task  $s$ ,
8:      $\mathbf{w}_s$  to clients allocated.
9:     Client performs local SGD (stochastic gradient descent)
    $\tau$  times for allocated task, optimizing  $\mathbf{w}_{k,s}$ , and uploads  $\mathbf{w}_{k,s}$  to
   the server.
10:    end for
11:   for all tasks  $s \in \mathbb{S}$  do
12:     Server aggregates  $\mathbf{w}_s \leftarrow \mathbf{w}_s - \eta_t \sum_k p_{k,Sel} \nabla F_k(\mathbf{w}_s)$ .
13:     Obtain global accuracy  $f_s$  for task  $s$ .
14:   end for
15: end for

```

---

accuracy, which intuitively improves fairness across tasks. Firstly, we denote  $\mathbf{w}_{s,\alpha}^* = G^\alpha = \min_{\mathbf{w}_s} g^\alpha(\{\mathbf{w}_s\})$ , where

$$G^\alpha = \min_{\mathbf{w}_s} \left\{ \left( \frac{1}{S} \sum_{s=1}^S f_s^\alpha(\mathbf{w}_s) \right), \text{s.t. } \sum_s a_{k,s} = 1 \forall k \forall t \right\}, \quad (5)$$

where  $a_{k,s}$  indicates that user  $k$  trains task  $s$  during epoch  $t$  and  $\sum_s a_{k,s} = 1 \forall k \forall t$  expresses the constraint that each client trains 1 task each epoch, and  $\mathbf{w}_{s,\alpha}^*$  is the global optimal solution when parameter  $\alpha$  is used. In other words,  $G^\alpha$  represents the optimal value of our MMFL objective (2).

To motivate the choice of  $\alpha$ , we show that for  $\alpha = 1, 2$ , a higher  $\alpha$  puts more emphasis on fairness: we relate this  $\alpha$ -fairness metric to the well-known *variance* and *cosine similarity* metrics for similarity across task accuracies.

**Lemma 1** (Fairness and Variance).  $\alpha = 2$  yields a smaller variance in the task accuracies than  $\alpha = 1$ :

$$\text{Var}(f_1(\mathbf{w}_{1,2}^*), \dots, f_n(\mathbf{w}_{n,2}^*)) \leq \text{Var}(f_1(\mathbf{w}_{1,1}^*), \dots, f_n(\mathbf{w}_{n,1}^*)).$$

*Proof Sketch:* We followed the outline of the proof if [17]. As we are interested in task fairness, unlike [17], we replace client loss functions with the global task loss functions  $f_s(\mathbf{w}_{s,\alpha})$  in our derivations. We next combine the definitions of the variance and  $\mathbf{w}_{s,\alpha}^*$  in Eq. (5) to obtain our result.  $\square$

Next, we use the definition of cosine similarity to show that  $\alpha = 2$  results in a fairer solution than  $\alpha = 1$  (non-fairness).

**Lemma 2** (Fairness and Cosine Similarity).  $\alpha = 2$  results in a higher cosine similarity across tasks than  $\alpha = 1$ , i.e.

$$\frac{\frac{1}{S} \sum_{s=1}^S f_s(\mathbf{w}_{s,2}^*)}{\sqrt{\frac{1}{S} \sum_{s=1}^S f_s^2(\mathbf{w}_{s,2}^*)}} \geq \frac{\frac{1}{S} \sum_{s=1}^S f_s(\mathbf{w}_{s,1}^*)}{\sqrt{\frac{1}{S} \sum_{s=1}^S f_s^2(\mathbf{w}_{s,1}^*)}}. \quad (6)$$

*Proof Sketch:* We followed the outline of [17]. Unlike [17], we are concerned with fairness across tasks, hence we replace user loss functions with the global task loss functions in our derivations. We further use the definition of cosine similarity, and the definition of  $\mathbf{w}_{s,\alpha}^*$  and  $G^\alpha(\mathbf{w}_s)$  to obtain our result.  $\square$

## B. Convergence Guarantees

Here, we analyze the convergence of training tasks under our client-task allocation algorithm, showing that it preferentially accelerates training of tasks with higher losses and that each task will converge. We prove convergence by taking the perspective of a single task in MMFL, without loss of generality.

We first characterize  $B_{Sel}^s$ , the **probability that a specific set of clients  $Sel$  is allocated** (i.e., selected) for a specific task  $s$  by our algorithm **FedFairMMFL**.

$$B_{Sel}^s(\alpha) = \left( \frac{f_s^\alpha(t)}{\sum_{s' \in S} f_{s'}^\alpha(t)} \right)^{|Sel|} \left( 1 - \frac{f_s^\alpha(t)}{\sum_{s' \in S} f_{s'}^\alpha(t)} \right)^{K-|Sel|} \quad (7)$$

$B_{Sel}^s(\alpha)$  depends on the losses of all tasks  $s \in S$ , and our  $\alpha$ -fair parameter  $\alpha$ . Next we introduce  $\Gamma_s$ , the *gap* between the loss evaluated at the global optimum  $\mathbf{w}_s^*$ , and the loss evaluated at client  $k$ 's optimal  $\mathbf{w}_{k,s}^*$ , for task  $s$ :  $\Gamma_s = \sum_{k=1}^K p_k (F_{k,s}(\mathbf{w}_s^*) - F_{k,s}(\mathbf{w}_{k,s}^*)) = f_s^* - \sum_{k=1}^K p_k F_{k,s}^*$ . We further characterise the *selection skew*  $\rho^\alpha(B_{Sel}^s, \mathbf{w}')$ . The selection skew is a ratio of the expected ‘local-global objective gap’ of the selected clients under **FedFairMMFL** over the ‘local-global objective gap’ of all  $K$  clients, where  $\mathbf{w}'$  is the point at which  $F_{k,s}$  and  $f_s$  are evaluated. This skew depends on the expectation of client allocation combinations, and hence on  $B_{Sel}^s$  and the  $\alpha$ -fair parameter  $\alpha$ :

$$\begin{aligned} & \rho^\alpha(B_{Sel}^s, \mathbf{w}') \\ &= \frac{\sum_{Sel \in \mathbb{S}} B_{Sel}^s(\alpha) \sum_{k \in Sel} \frac{p_k}{\sum_{k' \in Sel} p_{k'}} (F_{k,s}(\mathbf{w}') - F_{k,s}^*)}{f_s(\mathbf{w}') - \sum_{k=1}^K p_k F_{k,s}^*} \end{aligned}$$

We use  $\bar{\rho}^\alpha$  and  $\underline{\rho}^\alpha$  respectively to denote upper and lower bounds on the selection skew throughout the training. Recall that  $\tau$  denotes the number of SGD epochs in local training. We then introduce the following assumptions used in our convergence analysis, which are standard in the FL literature [22].

**Assumption 1:**  $F_{1,s}, \dots, F_{k,s}$  are  $L$ -smooth, i.e.  $F_{k,s}(\mathbf{v}) \leq F_{k,s}(\mathbf{w}) + (\mathbf{v} - \mathbf{w})^T \nabla F_{k,s}(\mathbf{w}) + \frac{L}{2} \|\mathbf{v} - \mathbf{w}\|^2$ ,  $\forall \mathbf{v}$  and  $\mathbf{w}$ .

**Assumption 2:**  $F_{1,s}, \dots, F_{k,s}$  are  $\mu$ -strongly convex, i.e.  $F_{k,s}(\mathbf{v}) \geq F_{k,s}(\mathbf{w}) + (\mathbf{v} - \mathbf{w})^T \nabla F_{k,s}(\mathbf{w}) + \frac{\mu}{2} \|\mathbf{v} - \mathbf{w}\|^2$ , for all  $\mathbf{v}$  and  $\mathbf{w}$ . Note that  $\mu \leq L$  from Assumption 1.

**Assumption 3:** For all clients  $k$ , for the minibatch  $\xi_{k,s}$  uniformly sampled at random, the stochastic gradient is unbiased:  $\mathbb{E}[g_{k,s}(\mathbf{w}_{k,s}, \xi_{k,s})] = \nabla F_{k,s}(\mathbf{w}_{k,s})$ , and its variance is bounded:  $\mathbb{E}\|g_{k,s}(\mathbf{w}_{k,s}, \xi_{k,s}) - \nabla F_{k,s}(\mathbf{w}_{k,s})\|^2 \leq \sigma^2$ .

**Assumption 4:** Bounded stochastic gradient. For all clients  $k$ , the expected square norm of the stochastic gradient is bounded  $\mathbb{E}\|g_{k,s}(\mathbf{w}_{k,s}, \xi_{k,s})\|^2 \leq G^2$ .

**Lemma 3** (Local and Global Model Discrepancy). *The expected average discrepancy between the global model for task  $s$ ,  $\bar{\mathbf{w}}_s^{(t)}$  and client  $k$ 's model for  $s$ ,  $\mathbf{w}_{k,s}^{(t)}$ , under selection*

$k \in Sel(t)$ , can be bounded as

$$\mathbb{E} \left[ \sum_{k \in Sel(t)} \frac{p_k}{\sum_{k' \in Sel(t)} p_{k'}} \|\bar{\mathbf{w}}_s^{(t)} - \mathbf{w}_{k,s}^{(t)}\|^2 \right] \quad (8)$$

$$\leq 16\eta_t^2 \tau^2 G^2 \sum_{Sel(t) \in \mathbb{S}(t)} B_{Sel(t)}^s(\alpha) P_{Sel(t)}^{k,2} \quad (9)$$

$$\text{where } P_{Sel(t)}^{k,2} = \sum_{\substack{k \neq k', \\ k, k' \in Sel(t)}} \frac{p_k}{\sum_{k'' \in Sel(t)} p_{k''}} \frac{p_{k'}}{\sum_{k'' \in Sel(t)} p_{k''}}.$$

In the next theorem, without loss of generality we show the convergence rate of a task in the MMFL setting under **FedFairMMFL**, and show its dependence on the relative loss levels of all tasks. When the prevailing loss level of task  $s$  is higher, its convergence will be sped up under our algorithm.

**Theorem 4** (Model Parameter Convergence). *Suppose we use learning rate  $\eta_t$  at each round  $t$ . Under Assumptions 1-4, the progress of task  $s$ 's global parameters at round  $t+1$ ,  $\bar{\mathbf{w}}_s^{(t+1)}$ , towards the optimal parameters  $\mathbf{w}^*$ :  $\mathbb{E}[\|\bar{\mathbf{w}}_s^{(t+1)} - \mathbf{w}_s^*\|^2]$  is upper-bounded by*

$$\begin{aligned} & \left[ 1 - \eta_t \mu \left( 1 + \frac{3\bar{\rho}^\alpha}{8} \right) \right] \mathbb{E}[\|\bar{\mathbf{w}}^{(t)} - \mathbf{w}_s^*\|^2] + 2\eta_t \Gamma_s (\bar{\rho}^\alpha - \underline{\rho}^\alpha) \\ & + \eta_t^2 \sigma^2 \sum_{Sel(t) \in \mathbb{S}(t)} \sum_{k \in Sel(t)} B_{Sel(t)}^s(\alpha) \left( \frac{p_k}{\sum_{k' \in Sel(t)} p_{k'}} \right)^2 \\ & + \eta_t^2 (32\tau^2 G^2 + 6\bar{\rho}^\alpha L \Gamma_s). \end{aligned} \quad (10)$$

*Proof Sketch:* We adapted the outline of [22]'s proof to incorporate our distinct client selection scheme, which unlike [22]'s proof requires explicitly quantifying the impact of  $Sel(t)$  and  $B_{Sel}^s$  (Eq. 7) through various bounds on the loss function gradients. In the rest of our proof, we add and subtract multiple expressions involving the gradients, bound the various terms, and incorporate Lemma 3's result.  $\square$

In tracking the impact of our client-task allocation on the convergence in Theorem 4's proof, we also reveal how the convergence bound in (10) depends on our client-task allocation algorithm, in particular the parameter  $\alpha$  and the current task losses. This result not only helps us to prove the convergence of our **FedFairMMFL** training algorithm, but also provides additional justification for our client-task allocation strategy: Mathematically, tasks that are further from the optimum (i.e., with a larger  $\mathbb{E}\|\bar{\mathbf{w}}^{(t)} - \mathbf{w}_s^*\|^2$ ) should be allocated clients to reduce the right-hand side of (10) to equalize the progress across all tasks, which agrees with our algorithm. In Corollary 5, we show that under certain conditions on  $p_k$ , as  $\alpha$  increases, the term with  $B_{Sel}^s(\alpha)$  decreases, hence controlling the task's progress towards its optimum.

**Corollary 5** (Fairness and Convergence). *Suppose that all of the  $p_k = \frac{1}{K}$  and consider task  $s$  that has the highest loss  $f_s(\mathbf{w}_s(t+1))$ . Then as  $\alpha$  increases, the term  $\eta_t^2 \sigma^2 \sum_{Sel \in \mathbb{S}(t)} \sum_{k \in Sel} B_{Sel}^s(\alpha) \left( \frac{p_k}{\sum_{k' \in Sel} p_{k'}} \right)^2$  in the right-hand side of (10) decreases.*

*Proof.* Let  $\bar{f}_s(\alpha) = \frac{f_s(\mathbf{w}_s(t+1))^\alpha}{\sum_{s'} f_{s'}(\mathbf{w}_s(t+1))^\alpha}$ . When  $p_k = \frac{1}{K}$ , we can rewrite  $\sum_{Sel \in \mathbb{S}(t)} \sum_{k \in Sel} B_{Sel}^s(\alpha) \left( \frac{p_k}{\sum_{k' \in Sel} p_{k'}} \right)^2$  as  $\sum_{j=1}^K \frac{1}{j} (\bar{f}_s(\alpha))^j (1 - \bar{f}_s(\alpha))^{K-j} \binom{K}{j}$ .

Since  $\sum_{j=1}^K (\bar{f}_s(\alpha))^j (1 - \bar{f}_s(\alpha))^{K-j} \binom{K}{j} = 1$ , we can view this as a weighted average of  $1, \frac{1}{2}, \dots, \frac{1}{K}$ . It then suffices to show that, as  $\alpha$  increases, we place more weight on smaller terms. Note that  $\frac{d}{d\alpha} \left( (\bar{f}_s(\alpha))^j (1 - \bar{f}_s(\alpha))^{K-j} \right) \propto \left( \frac{d\bar{f}_s(\alpha)}{\alpha} \right) (j - K\bar{f}_s(\alpha))$ . It then suffices to show that  $\frac{d\bar{f}_s(\alpha)}{\alpha} > 0$ ; if so, the terms with larger  $j$  will increase as  $\alpha$  increases. Hence,  $\frac{d\bar{f}_s(\alpha)}{\alpha} \propto \sum_{s'=1}^S f_{s'}^\alpha (\log f_s - \log f_{s'}) > 0$  as we have assumed task  $s$  has the highest loss  $f_s$ .  $\square$

As the fairness parameter  $\alpha$  increases, our client-task allocation algorithm ensures that the worst-performing task makes more progress towards its training goal, thus helping to equalize the tasks' performance and fulfilling the intuition of a higher  $\alpha$  leading to more fair outcomes. The result further suggests that an alternative client-task allocation algorithm can explicitly enforce fairness across the upper-bounds in (10). However, this approach requires integer programming methods and estimating unknown constants like  $\sigma^2$  for each task, making it less practical than **FedFairMMFL**. Finally, we provide the convergence bound for **FedFairMMFL**:

**Corollary 6** (Model Convergence). *With Assumptions 1-4, under a decaying learning rate  $\eta_t = \frac{1}{\mu(t+\gamma)}$  and under our difficulty-aware client-task allocation algorithm, the error after  $T$  iterations of federated learning for a specific task  $s$  in multiple-model federated learning is*

$$\begin{aligned} \mathbb{E}[f_s(\bar{\mathbf{w}}_s^{(T)})] - f_s^* &\leq \frac{1}{T+\gamma} \left[ \frac{4[16\tau^2 G^2 + \sigma^2]}{3\tilde{\rho}^\alpha \mu^2} + \frac{8L^2 \Gamma_s}{\mu^2} \right. \\ &\quad \left. + \frac{L\gamma \|\bar{\mathbf{w}}_s^{(0)} - \mathbf{w}_s^*\|^2}{2} \right] + \frac{8L\Gamma_s}{3\mu} \left( \frac{\tilde{\rho}^\alpha}{\rho^\alpha} - 1 \right). \end{aligned}$$

*Proof Sketch.* We follow the outline of the proof of [22]. As our client selection scheme is different from that in [22], we use the upper bound of  $\mathbb{E}[\|\bar{\mathbf{w}}_s^{(t+1)} - \mathbf{w}_s^*\|^2]$  from Theorem 4, which incorporates our client-task allocation scheme. Next, we upper bound  $\Delta_{t+1} = \sum_{Sel \in \mathbb{S}} B_{Sel}^s(\alpha) P_{Sel}^{k,2}$  by  $\frac{1}{2}$ . Firstly, the inner summation  $P_{Sel}^{k,2}$  can be re-written as  $\sum_{k \in Sel} \sum_{k' \in Sel, k' \neq k} \frac{p_k}{(\sum_{k'' \in Sel} p_{k''})} \frac{p_{k'}}{(\sum_{k'' \in Sel} p_{k''})} = \frac{(\sum_{k'' \in Sel} p_{k''})^2 - \sum_{k \in Sel} p_k^2}{2(\sum_{k'' \in Sel} p_{k''})^2} \leq \frac{1}{2}$ . Secondly, the outer summation  $\sum_{Sel \in \mathbb{S}} B_{Sel}^s(\alpha)$  is a binomial distribution with binomial parameter  $f_s^\beta(t)/\sum_{s' \in S} f_{s'}^\beta(t)$ , and this sums to 1. Now we have a bound of  $\Delta_{t+1}$  in terms of  $\Delta_t$  and the learning rates  $\eta_t$ . Next, we use induction to bound  $\Delta_{t+1}$  in terms of time independent variables. Finally, we use the L-smoothness of  $F_s$  to obtain the result of this corollary.  $\square$

## V. CLIENT INCENTIVE-AWARE FAIRNESS ACROSS TASKS

In this section, we account for the fact that *clients may have different willingness to train each model*. Prior incentive-aware

FL work [31] aimed to maximize the total number of users performing either one (or many) tasks. Here, we would like to ensure a “fair” number of users incentivized across tasks, to further ensure fair training outcomes. To avoid excessive overhead during training, this client recruitment occurs before training begins [37]. In this section, we refer to potential clients as ‘users’, as they have not yet committed to the training. We model user  $i$ 's unwillingness towards training task  $s$  by supposing  $i$  will not train task  $s$  unless they receive a payment  $\geq c_{i,s}$ . Following prior work [15], we suppose the  $c_{i,s}$  are private and employ an auction that does not require users to explicitly report  $c_{i,s}$ ; auctions require no prior information about users' valuations, which may be difficult to obtain in practice. Users will submit bids  $b_{i,s}$  indicating the payment they want for each task. Given the bids, the server will choose the auction winners (which users will participate in training) and each user's payment. User  $i$ 's utility for task  $s$  will be

$$u_{i,s} = \begin{cases} p_{i,s} - c_{i,s} & \text{if they participate in this task} \\ 0 & \text{otherwise} \end{cases} \quad (11)$$

where  $p_{i,s}$  is the payment received. We suppose that there is a shared budget  $B$  across tasks. Since  $B$  is unlikely to be sufficient to incentivize all users to perform all tasks, we seek to distribute the budget so as to ensure fair training across the tasks. We consider two types of “fairness”: *budget-fairness*, i.e., an equal budget for each task; and *max-min fairness*, distributing the budget so as to maximize the minimum take-up rate across tasks. Budget fairness seeks to equalize the resources (in terms of budget) across task, while max-min fairness seeks to make the training outcomes fair.

### A. Fairness With Respect to the Same Budget across Tasks

Here, fairness is with respect to an equal budget  $B$  across tasks. We aim to maximize the total take-up rate across tasks:

$$\max_{\mathbf{x}_s \in \{0,1\}^I, \mathbf{p}} \sum_{s \in S} \left( \sum_{i \in I} x_{i,s} \right) \quad (12)$$

$$\text{s.t. } \sum_{i \in I} p_{i,s} x_{i,s} \leq \frac{B}{S}, \forall s \text{ and } p_{i,s} \geq b_{i,s} \mathbb{1}_{\{x_{i,s} \geq 0\}}, \forall i. \quad (13)$$

The decision variables for the server/ training coordinator are  $\mathbf{x}_s$ , a binary vector indicating whether each user is incentivized for task  $s$ , and  $\mathbf{p}$ , the set of payments to each user. The first constraint is the budget constraint for each task, and the second is that the user  $i$  agrees to train task  $s$  if  $u_{i,s} \geq 0$ .

We can decouple (12) across tasks and thus consider each task separately. The Proportional Share Mechanism proposed in [38] solves this problem by designing a mechanism that incentivizes users to report truthful bids  $b_{i,s} = c_{i,s}$ : after ordering bids in ascending order, for  $i = 1, 2, \dots$ , check if  $b_{i,s} \leq \frac{B}{Si}$ . Find the smallest  $k$  s.t.  $b_{k,s} > \frac{B}{Sk}$ . This bid  $b_{k,s}$  is the first loser, not in the winning set. All bids smaller than  $b_{k,s}$  will be in the winning set and are paid  $\frac{B}{S(k-1)}$ .

While this budget-fair auction is truthful, tasks disfavored by users (i.e., for which all users have high  $c_{i,s}$ ) may still

---

**Algorithm 2** GMMFair: Greedy max-min fair algorithm

---

```

1: Initialization:  $S_s = \{\}$   $\forall s$ , each task  $s$  has no users
2: for task  $s = 1, 2, \dots, S$  do
3:   Order bids for task  $s$  in ascending order
4: end for
5:  $t = 1$ . Let  $b_{t,s}$  denote the  $t$ -th smallest bid.
6: while  $\sum_s b_{t,s} \leq B$ , Check if there is enough budget to add one
   more user to each task do
7:   for task  $s = 1, 2, \dots, S$  do
8:      $S_s \leftarrow S_s \cup t$ ; add user with minimum bid to  $S_s$ 
9:      $B \leftarrow B - b_t$ , update remaining budget
10:  end for
11:   $t \leftarrow t + 1$ 
12: end while

```

---

be able to incentivize few clients, leading to poor training performance. Thus, we next look at max-min fairness.

### B. Fairness with Respect to Max-Min Task Take-Up Rates

We aim to maximize the minimum client take-up rate across tasks  $\min_{s \in S} (\sum_{i \in I} x_{i,s})$ , in light of the user valuations  $c_{i,s}$ .

$$\max_{\mathbf{x}_s \in \{0,1\}, \mathbf{p}} \min_{s \in S} \left( \sum_{i \in I} x_{i,s} \right) \text{ s.t. } \sum_{s \in S} \sum_{i \in I} p_{i,s} x_{i,s} \leq B, p_{i,s} \geq b_{i,s} \quad (14)$$

As in (12)'s formulation, the decision variables are  $\mathbf{x}_s$ , and  $\mathbf{p}$ . The first constraint is the budget constraint, where the budget  $B$  is coupled across the tasks  $s \in S$ . The second constraint is that users' payments must exceed their bids. We first provide an algorithm to solve (14) in Algorithm 2, which greedily adds a user to each task until the budget  $B$  is exhausted.

**Lemma 7** (Max-Min Fair Solution Algorithm). *Algorithm 2 solves the optimization problem (14).*

*Proof sketch.* For an optimal max-min objective value  $t$ , we show by contradiction that all users with bids  $\leq b_t$  must be included in the optimal set. Algorithm 2 finds this solution. See the Technical Report [36] for the full proof.  $\square$ .

While Algorithm 2 is optimal, it is not truthful: any user  $i$  in the winning set for a task  $s$  can easily increase its payment by increasing its bid  $b_{i,s} > c_{i,s}$ . Thus, we next design an auction method that, like Algorithm 2, proceeds in rounds to add an additional user to each task. We adapt the payment rule from the budget-fair auction in Section V-A to help ensure truthfulness. Our novel **max-min fair auction** is presented in **Algorithm 3**. The main idea, is to begin with a budget-fair auction for each task, and allocate users to tasks in rounds until a task runs out of budget (lines 4-9), as in Algorithm 2. We then re-allocate the budget across tasks so as to keep allocating users to all tasks, if possible (lines 11-16). If there is insufficient budget to do so, we can optionally perform a *fractional* allocation that uses up the remaining budget while remaining fair (lines 17-23). This ensures max-min fairness, and that the difference in number of users allocated across tasks is at most a fraction. In the learning process, this fraction can be modelled in terms of the time spent training tasks.

---

**Algorithm 3** MMFL Max-min Fair

---

```

1: Initialization:  $B_s^0 \leftarrow \frac{B}{s}$ , equal initial budget  $\forall$  tasks.
2: For all tasks, order received bids  $b_i$  in ascending order.
3: for round  $i = 1, 2, \dots$  do
4:   for all tasks  $s$  do
5:     Let  $\underline{b}_{i,s}$  denote the  $i$ -th biggest bid.
6:     if  $\underline{b}_{i,s} \leq \frac{B}{i}$  then
7:       User joins the winners' set
8:       and the Payment is updated  $p_{i,s} \leftarrow \frac{B^t}{t}$ .
9:     end if
10:    end for
11:    if  $\exists i \geq 1$  task s.t. the  $i$ -th user does not join then
12:      Let  $S_{i-1}$  and  $S_i$  denote the set of tasks with  $i-1$  and
          $i$  users respectively.
13:       $A = \sum_{s \in S_{i-1}} [\underline{b}_{i,s} i - B_s^{i-1}]$ 
14:       $C = \sum_{s \in S_i} [B_s^{i-1} - \underline{b}_{i,s} i]$ 
15:      if  $A < C$  then
16:        Re-allocate amount  $A$  from  $S_i$  to  $S_{i-1}$ , obtaining  $A$ 
         from  $s \in S_i$  via a waterfilling manner.
17:      else
18:        Fractionally allocate  $C$  across  $S_{i-1}$ .
19:         $p_{i,s} = \begin{cases} \frac{C}{|S_{i-1}|} & \text{if } \frac{C}{|S_{i-1}|} \leq \underline{b}_{i,s} \\ \underline{b}_{i,s} & \text{otherwise} \end{cases}$ 
20:         $x_{i,s} = \begin{cases} \frac{C}{|S_{i-1}| \underline{b}_{i,s}} & \text{if } \frac{C}{|S_{i-1}|} \leq \underline{b}_{i,s} \\ 1 & \text{otherwise} \end{cases}$ 
21:        End the Auction.
22:      end if
23:    end if
24:  end for

```

---

Without the re-allocation across tasks, we would expect this auction to be truthful. Indeed, we show that such re-allocation rounds are the only instances that incentivize un-truthfulness:

**Theorem 8** (Untruthfulness Conditions). *The truthful strategy dominates, except in the following cases where user  $i$  obtains higher utility by bidding  $\hat{b}_{i,s} > c_{i,s}$ . In all cases,  $\nexists b_{j,s} < \hat{b}_{i,s}$  for which a)  $b_{j,s} + \sum_{s' \in S_{j-1}} b_{j,s'} > C$  and b)  $b_{j,s} > \frac{B^j}{j}$ .*

Case 1 c)  $\hat{b}_{i,s} > \frac{B^t}{t}$  and d)  $\hat{b}_{i,s} + \sum_{s' \in S_{t-1}} b_{t,s'} < C$ .

Case 2 If there is no  $b_{j,s}$ ,  $b_{i,s} < b_{j,s} < \hat{b}_{i,s}$ , and no  $b_{j,s}$ ,  $\underline{b}_{j,s} < b_{i,s} < \hat{b}_{i,s}$  for which e)  $b_{i,s} > \frac{B^t}{t}$  f)  $b_{i,s} + \sum_{s' \in S_{t-1}} b_{t,s'} > C$ , g)  $\hat{b}_{i,s} > \frac{B^t}{t}$ , h)  $\hat{b}_{i,s} + \sum_{s' \in S_{t-1}} b_{t,s'} > C$ , and i)  $\min(b_{i,s}, \frac{C}{|S_{t-1}|}) < \min(\hat{b}_{i,s}, \frac{C}{|S_{t-1}|})$ .

Case 3 If there is no  $b_{j,s}$ ,  $b_{i,s} < b_{j,s} < \hat{b}_{i,s}$  for which j)  $\hat{b}_{i,s} + \sum_{s' \in S_{t-1}} b_{t,s'} > C$ : k) (there exists  $l < t$  s.t.  $b_{i,s} < \frac{B^l}{l}$ ), l)  $\hat{b}_{i,s} > \frac{B^t}{t}$ , and m)  $\hat{b}_{i,s} < \frac{C}{|S_{t-1}|}$  or  $\frac{C}{|S_{t-1}|} > \frac{B^t}{t}$ .

Case 4 If there is no  $b_{j,s}$ ,  $b_{i,s} < b_{j,s} < \hat{b}_{i,s}$ , and no  $b_{j,s}$ ,  $\underline{b}_{j,s} < b_{i,s} < \hat{b}_{i,s}$  for which n)  $b_{i,s} > \frac{B^t}{t}$  o)  $b_{i,s} + \sum_{s' \in S_{t-1}} b_{t,s'} < C$ , p)  $\hat{b}_{i,s} > \frac{B^t}{t}$ , and q)  $\hat{b}_{i,s} + \sum_{s' \in S_{t-1}} b_{t,s'} > C$ , r)  $b_{i,s} < \min(\frac{C}{|S_{t-1}|}, \hat{b}_{i,s})$ .

*Proof Sketch:* We characterize the untruthful cases (all during re-allocation rounds). The negation of the conditions a) and b) indicate that the auction does not end at other bids

$\underline{b}_{j,s}$  smaller than the untruthful bid  $\hat{b}_{i,s}$ . We analyze all cases in detail in the Technical Report [36].  $\square$

In practice, it would be *difficult for users to exploit these conditions*: all four conditions rely on the un-truthful bid comprising part of a re-allocation round. This heavily depends on other users' bids, which would not be known. Moreover, since re-allocations are more likely to happen near the end of the auction, such un-truthful bids have a high risk of not being in the winning set at all. Next, we show that even if a user successfully submits such an un-truthful bid, it would have limited impact on the max-min fairness objective:

**Corollary 9** (Untruthfulness Impact). *Suppose a user submits an un-truthful bid under one of Theorem 8's conditions. Under the un-truthful setting, the fractional re-allocation round will arrive at most one round earlier, compared to the truthful setting. This reduces the max-min objective by at most 2.*

*Proof Sketch:* The proof follows from analyzing each case in Theorem 8, as detailed in the Technical Report ([36]).  $\square$

We now compare our max-min fair auction to the (truthful) budget-fair auction. Intuitively, we expect that it may lead to more fair outcomes. We can quantify this effect by studying the extreme case in which one task has no users, even if user bids from two tasks follow the same distribution:

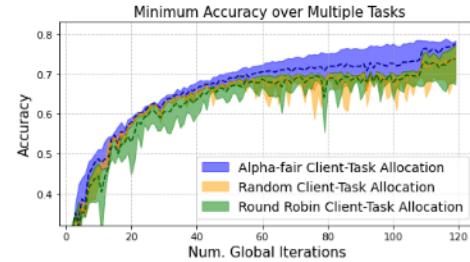
**Corollary 10** (Auction Comparison). *Suppose that all user bids for a given task  $s$  are independent samples from a probability distribution  $f_s$ . Let  $x_s \sim f_s$  denote the minimum of the  $K$  user bids for task  $s$ . Then the max-min fair auction has a smaller probability that at least one task is allocated no users compared to the budget-fair auction:  $\mathbb{P}[\sum_{s=1}^S x_s \geq B] \leq \mathbb{P}[\max_{s=1}^S x_s \geq \frac{B}{S}]$ . For example, if  $f_s \sim \exp(\lambda)$  and  $S = 2$ , then the probability at least one task is allocated no users in the max-min fair auction is  $\exp(-B\lambda)(1 + B\lambda)$ , which is strictly greater than the probability at least one task is allocated no users in the budget-fair auction,  $\exp(-B\lambda)(2 \exp(\frac{B\lambda}{2}) - 1)$  if  $\lambda B > 0$ .*

*Proof.* Under the max-min fair auction, a given task will have no users allocated if the sum of the lowest bids for each task is larger than the budget  $B$ . Similarly, under the budget-fair auction, a given task will have no users allocated if its lowest bid exceeds its budget  $B/S$ . When  $f_s \sim \exp(\lambda)$  and  $S = 2$ ,  $x_1 + x_2 \sim \gamma(2, 1/\lambda)$ , and  $\max(x_1, x_2) \sim (1 - \exp(-\lambda x))^2$ . We can then show  $(1 - \sum_{s=0}^1 (B\lambda)^s \exp(-B\lambda)) \geq (1 - \exp(-\frac{B\lambda}{2}))^2$ .  $\square$

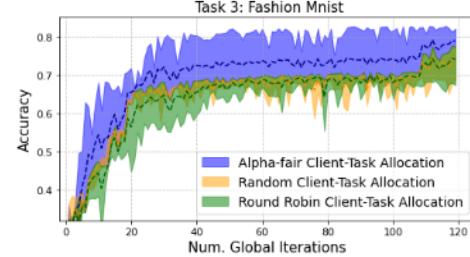
### C. Convergence Guarantees of our Incentive-Aware Auctions

We provide another convergence bound, based on the results of our incentive-aware auctions. For this, we firstly characterise the probabilities of users being in the auctions' winning set (i.e., probabilities they agree to train a particular task).

Under the Budget-Fair Auction, the probability  $p_{B,b_i,s}^{join}$  that a user with bid  $b_i$  joins task  $s$ , given a total budget of  $B$  is  $\sum_{\bar{t}} P(b_{i,s} \text{ is } \bar{t}\text{-th smallest})P(\frac{B}{s\bar{t}} > b_{i,s})$ . Under the Max-min Fair Auction, the probability



(a) Minimum accuracy across the 3 tasks.



(b) Accuracy for task 3: Fashion-MNIST.

**Fig. 2. Results of Experiment 1: 3 tasks with varying difficulty levels.** The 3 tasks have datasets MNIST, CIFAR-10, and Fashion-MNIST. **FedFair-MMFL** achieves a higher minimum accuracy across tasks, as compared to the baselines, while maintaining the same or higher accuracy for the other 2 tasks (see Technical Report [36]).

$p_{B,b_i,s}^{join}$  is  $\sum_{\bar{t}} P(b_{i,s} \text{ is } \bar{t}\text{-th smallest})P(\frac{B}{s\bar{t}} > b_{i,s}) + \sum_{\hat{t}} P(b_{i,s} \text{ is } \hat{t}\text{-th smallest})P(\frac{B}{s\hat{t}} < b_{i,s})P(\sum_{s \in S_{\bar{t}-1}} b_{\bar{t},s} < C)$  for full participation, and  $\sum_{\bar{t}} P(b_{i,s} \text{ is } \bar{t}\text{-th smallest})P(\frac{B}{s\bar{t}} < b_{i,s}) \times P(\sum_{s \in S_{\bar{t}-1}} b_{\bar{t},s} > C) \min(1, \frac{C}{|S_{\bar{t}-1}|b_{i,s}})$  for partial participation. Letting  $\mathbb{W}$  be the set of auction winners, and  $W_K$  be all the subsets of  $K$  clients recruited from  $N$  users:

**Theorem 11** (Convergence with Auctions). *With Assumptions 1-4, under a decaying learning rate  $\eta_t = \frac{1}{\mu(t+\gamma)}$ , under an auction scheme and our difficulty-aware client-task allocation algorithm, the error after  $T$  iterations of federated learning for a specific task  $s$  in MMFL,  $\mathbb{E}[f_s(\bar{\mathbf{w}}_s^{(T)})] - f_s^*$ , is upper-bounded by*

$$\frac{1}{T+\gamma} \left[ [64\tau^2 G^2 \sum_{K=1}^N \sum_{\mathbb{W} \in W_K} J_{\mathbb{W}} + 4\sigma^2] / (3\bar{\rho}^\alpha \mu^2) + \frac{8L^2 \Gamma_s}{\mu^2} + \frac{L\gamma \|\bar{\mathbf{w}}_s^{(0)} - \mathbf{w}_s^*\|^2}{2} \right] + \frac{8L\Gamma_s}{3\mu} \left( \frac{\tilde{\rho}^\alpha}{\bar{\rho}^\alpha} - 1 \right)$$

where  $J_{\mathbb{W}} = \prod_{i \in \mathbb{W}} (p_{B,b_i}^{join}) \prod_{j \notin \mathbb{W}} (1 - p_{B,b_j}^{join})$ . Note that  $p_{B,b_i}^{join}$  will be modified depending on the auction used.

*Proof Sketch.* We use a Poisson binomial distribution to model users being in the winning set of the auction. We then follow the proof of Corollary 6.  $\square$

## VI. EVALUATION

Finally we evaluate **FedFairMMFL** and our auction schemes. Our **main results** are that (i) **FedFairMMFL** consistently performs fairly across different experiment settings and (ii) our max-min auction increases the minimum learning

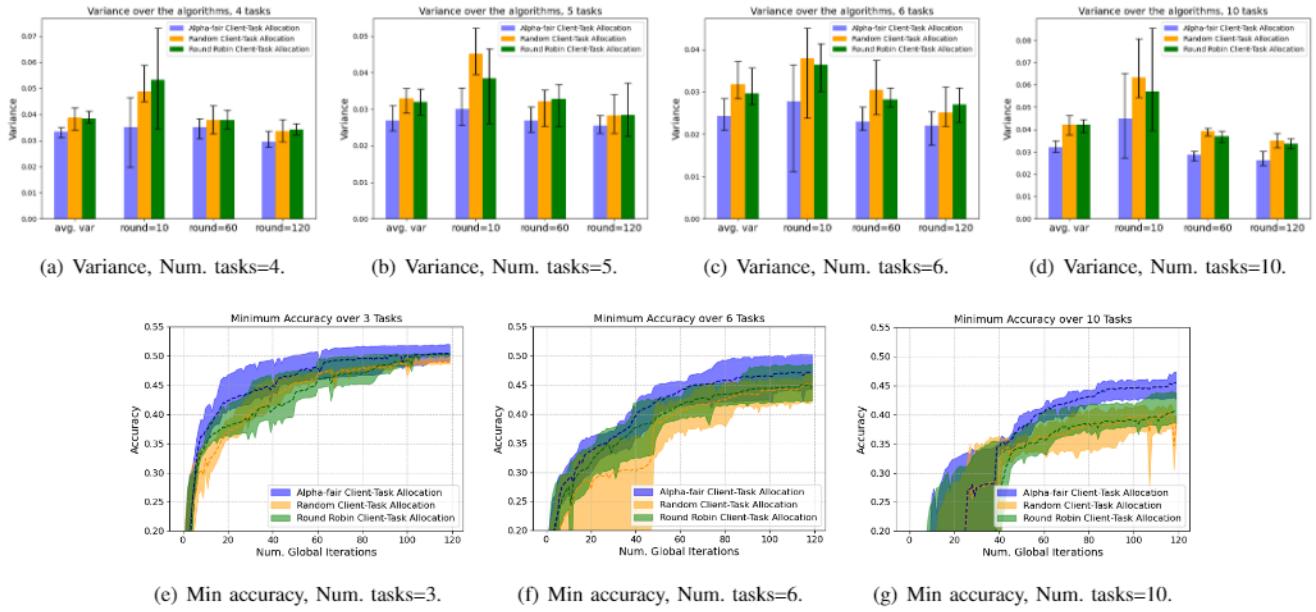


Fig. 3. Results of Experiment 2: Increasing the number of tasks. (a), (b), (c), and (d) compare the variance in accuracy observed across multiple tasks, when employing the different algorithms. The error bars denote the max/min values of the corresponding variance among all 4 random seeds. Our algorithm generally achieves a lower variance amongst task performance. (e), (f), and (g) show the trend of minimum accuracy when task numbers are 3, 6, and 10, respectively.

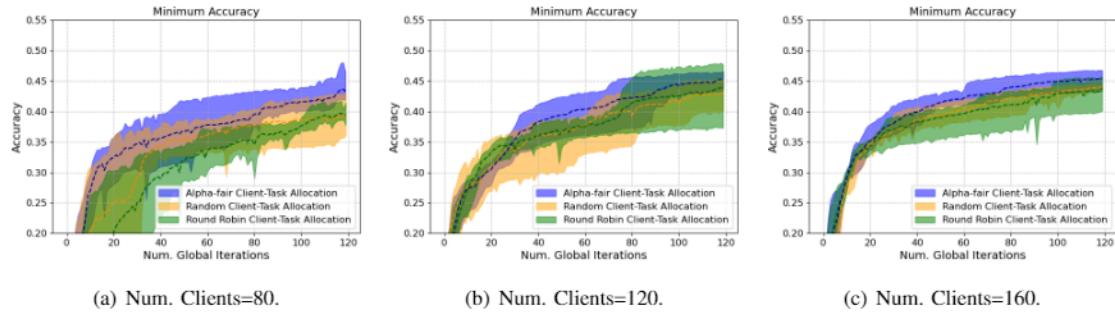


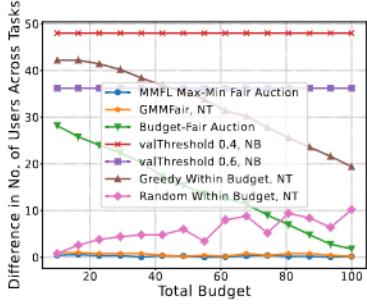
Fig. 4. Results of Experiment 3: Increasing the number of clients. All experiments have 5 tasks (MNIST, CIFAR-10, Fashion-MNIST, EMNIST, and CIFAR-10). Experiments are conducted with 4 random seeds. The participation rate is 0.25, and client numbers are 80, 120, and 160, respectively. Our algorithm FedFairMMFL generally achieves a higher minimum accuracy, and/or a faster convergence.

accuracy across tasks in the budget constrained setting. We concurrently train (i) three tasks of *varied* difficulty levels, and evaluate the algorithms' performance across varied numbers of (ii) tasks and (iii) clients. We analyze (iv) the impact of our auctions and baselines on the take-up rate over tasks, and analyze (v) the impact of these take-up rates on the tasks' learning outcomes. We use  $\alpha = 3$  for FedFairMMFL in all experiments. The effect of different  $\alpha$  is also measured (see Technical Report [36]).

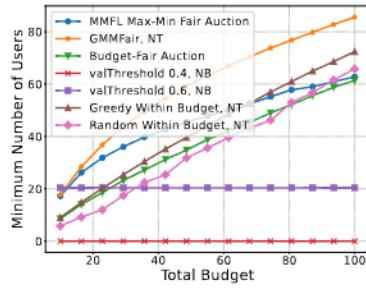
Our client-task allocation algorithm **FedFairMMFL** is measured against two **baselines**: a) *Random*, where clients are randomly allocated a task with equal probability each global epoch (note that this corresponds to taking  $\alpha = 1$  in our  $\alpha$ -fair allocation), and b) *Round Robin* [8], where active clients are allocated to tasks in a round-robin manner, sequentially across global epochs. Our main fairness **metrics** are the minimum test accuracy across all tasks and the variance of the test accuracies (cf. Lemma 1). As some models may have larger training losses by nature, e.g., if they are defined on different datasets,

making them more easily favored by the algorithm, we use the test accuracy instead of training loss in **FedFairMMFL**. Each experiment's results are averaged over 3-5 random seeds - we plot the average values (dotted line), along with the maximum and minimum values indicated by the shaded area. Each client's data is sampled from either all classes (i.i.d.) or a randomly chosen half of the classes (non-i.i.d.).

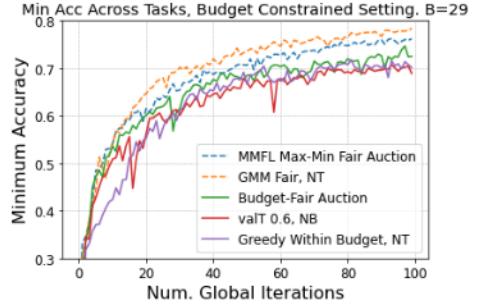
**Experiment 1: Tasks of Varying Difficulty Levels.** To model differing task difficulties, our tasks are to train image recognition models on MNIST and Fashion-MNIST on a convolutional neural network with 2 convolutional, 2 pooling, and 2 fully connected layers; and CIFAR-10 with a pre-activated ResNet. Experiments are run with 120 clients having 150–250 datapoints from five (randomly chosen) of the ten classes (i.e., non i.i.d. data); 35% of clients, chosen uniformly at random, are active each round. As seen in Fig. 2 a), **FedFairMMFL** has a higher minimum accuracy than the random and round robin baselines. **FedFairMMFL** adaptively allocates more clients to the prevailing worst-performing task, resulting in a higher



(a) Difference in the number of users which will train the tasks in MMFL.



(b) Minimum number of users recruited for training, over two tasks.



(c) The minimum accuracy after combining our auction strategies and baselines, with FedFairMMFL.

**Fig. 5. Results of Experiment 4: Client Take-Up Rates.** (NB= No Budget, NT= Non Truthful Mechanism). Our near-truthful auction MMFL Max-Min Fair (a) leads to a lower difference in the number of users across tasks, and (b) incentivizes more clients than any incentivization algorithm but the un-truthful GMMFair. **Results of Experiments 5: How auctions impact learning.** Fig. (c) shows the minimum accuracy after combining the results of our auction strategies and baselines, with FedFairMMFL. The MMFL Max-Min Fair Auction leads to higher accuracies.

accuracy for this task (Task 3: Fashion-MNIST), as seen in Fig. 2 b), while maintaining the same or a slightly higher accuracy for the other tasks (see Tech Report [36]). Though we might guess that CIFAR-10 would be the worst-performing task, as it uses the largest model, **FedFairMMFL** identifies that Fashion-MNIST is the worst-performing task, and it achieves fair convergence even in this challenging setting where Fashion-MNIST persistently performs worse than the other tasks.

**Experiment 2: Increasing the number of tasks.** To further understand the various algorithms (**FedFairMMFL** and the random allocation and round robin baselines) under different settings, their performance is measured as we increase the number of tasks, from 3 to 4, 5, 6 and 10. The tasks include MNIST, Fashion-MNIST, CIFAR-10, and EMNIST. We use the same networks as in Experiment 1 and a similar convolutional neural network with 2 convolutional, 2 pooling, and 2 fully connected layers for the EMNIST task. Experiments are run with 20 clients, having a participation rate of 1. Each client has 400 – 600 datapoints from each dataset for each task. We plot the variance amongst tasks on average, as well as at global epoch round numbers 10, 60, and 120. As seen in Fig. 3, with the increase in the round number, all algorithms’ variance decreases, and **FedFairMMFL** has the lowest variance for all settings. This can be explained through the fact that **FedFairMMFL** dynamically allocates clients with a higher probability to tasks which are currently performing worse, eliminating the difference among tasks as training progresses. This effect remains even as we increase the number of tasks, which slows the training for all tasks as the clients are being shared among more tasks - see Fig. 3 (e), (f), and (g), which compare the minimum accuracy among different algorithms when task numbers are 3, 6, and 10. **FedFairMMFL** converges faster than both baselines.

**Experiment 3: Increasing the number of clients.** Different client numbers are also tested to further evaluate **FedFairMMFL**’s performance. Here, all experiments have 5 tasks (MNIST, CIFAR-10, Fashion-MNIST, EMNIST, and CIFAR-10), with client participation rate 25% and 200–300 datapoints for each client. We vary the number of clients from 80 to

160. Experiment results are shown in Fig. 4. It is clear to see that in this setting, **FedFairMMFL** outperforms the other two algorithms, either achieving a higher minimum accuracy, and/or a faster convergence. Its performance is also further improved as the client number increases, and is more stable among different random seeds, likely because having more clients allows the worse-performing tasks to “catch up” with the others faster under the  $\alpha$ -fair allocation.

**Experiment 4: Client Take-Up Rates.** We compare the number of clients incentivized to train each task under our budget-fair (Section V-A) and max-min fair (Section V-B) auctions, as well as the (un-truthful) GMMFair algorithm, which upper-bounds the max-min fair auction result, and 4 other baselines: *valThreshold 0.4* and *valThreshold 0.6* assume no budget but that all users with cost (valuation)  $c_{i,s} < 0.4$  (respectively  $c_{i,s} < 0.6$ ) will agree to train the task; these simulate posted-price mechanisms that show the effect of the incentivization budget. *Greedy Within Budget* and *Random Within Budget* evenly divide the budget across tasks. Clients are added by ascending bid (for *Greedy Within Budget*) and randomly (for *Random Within Budget*), until the budget is consumed; comparison to these baselines shows the cost of enforcing truthfulness in the budget-fair auction. We consider two tasks and 100 clients; client bids are between 0 and 1 and are independently drawn from a truncated Gaussian distribution for Task 1 and an increasing linear distribution for Task 2. Results are averaged across 5 random seeds.

As seen in Fig. 5 a), our auction MMFL Max-min Fair leads to the lowest difference in the number of clients across tasks, managing to hit the theoretical optimal of the non-truthful algorithm GMMFair. *Greedy within budget* assigned the most clients to task 1, resulting in a high “difference” as it assigns less clients to task 2 (increasing linear distribution of valuations), nevertheless *Greedy within budget* (along with GMMFair and *Random within budget*) are not truthful. Both *Greedy within budget* (non truthful) and the Budget-Fair Auction become more fair (decreasing difference across tasks) as the budget increases. Nevertheless MMFL Max-Min Fair outperforms them in the budget constrained region. As seen

in Fig. 5 b), GMMFair yields the highest minimum number of users assigned to each task, consistent with Lemma 7. When the budget is low (*budget constrained regime*), our near-truthful auction MMFL Max-min Fair outperforms all the other mechanisms in the minimum number of users. As the budget increases, the greedy-based mechanisms outperform it. Nevertheless, they and GMMFair are not truthful, and can be manipulated if all users jointly submit high bids.

**Experiment 5: Take-Up Rates and Learning Outcomes.** We finally consider a constrained budget ( $B = 29$ ) and evaluate how the outcome of the client recruitment strategies impact MMFL model training. Fig. 5 c) shows the minimum test accuracy across the two tasks (MNIST and CIFAR-10) at each training round, using **FedFairMMFL** for client-task allocation. GMMFair consistently has the highest minimum accuracy across tasks, likely because it recruits the most clients to each task, as shown in Experiment 4 (Fig. 5 b). Our MMFL auction, which additionally has the benefit of near-truthfulness, achieves nearly the same minimum accuracy, showing that it performs well in tandem with **FedFairMMFL**.

## VII. CONCLUSION AND FUTURE DIRECTIONS

In this paper, we consider the fair training of concurrent MMFL tasks. Ours is the first work to both theoretically and experimentally validate algorithms that ensure fair outcomes across different learning tasks. We first propose **FedFair-MMFL**, a simple algorithm to allocate clients to tasks in each training round, showing that it permits theoretical convergence guarantees. We then propose two auction mechanisms, **Budget-Fair** and **MMFL Max-Min Fair**, that seek to incentivize a fair number of clients to commit to training each task. Our experimental results for training different combinations of FL tasks show that **FedFairMMFL** and **MMFL Max-Min Fair** can jointly ensure fair convergence rates and final test accuracies across the different tasks. As the first work to consider fairness in the MMFL setting, we expect our work to seed many future directions, including how to balance achieving fairness over tasks with fairness over clients' utilities. Another direction is to investigate fairness when tasks have differing difficulties and clients have heterogeneous, stochastic available computing resources for training each task.

## REFERENCES

- [1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.
- [2] Y. Yao, M. M. Kamani, Z. Cheng, L. Chen, C. Joe-Wong, and T. Liu, "Fedrule: Federated rule recommendation system with graph neural networks," in *Accepted to ACM/IEEE IoTDI*, 2023.
- [3] U. M. Aïvodji, S. Gambs, and A. Martin, "Iotfla: A secured and privacy-preserving smart home architecture implementing federated learning," in *2019 IEEE security and privacy workshops*. IEEE, 2019, pp. 175–180.
- [4] Y. Liu, L. Su, C. Joe-Wong, S. Ioannidis, E. Yeh, and M. Siew, "Cache-enabled federated learning systems," in *Proceedings of the Twenty-fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2023, pp. 1–11.
- [5] D. C. Nguyen, Q.-V. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, O. Dobre, and W.-J. Hwang, "Federated learning for smart healthcare: A survey," *ACM Computing Surveys*, vol. 55, no. 3, pp. 1–37, 2022.
- [6] A. Imteaj and M. H. Amini, "Distributed sensing using smart end-user devices: Pathway to federated learning for autonomous iot," in *2019 International conference on computational science and computational intelligence (CSCI)*. IEEE, 2019, pp. 1156–1161.
- [7] N. Bhuyan, S. Moharir, and G. Joshi, "Multi-model federated learning with provable guarantees," *arXiv preprint arXiv:2207.04330*, 2022.
- [8] N. Bhuyan and S. Moharir, "Multi-model federated learning," in *2022 14th International Conference on COMmunication Systems & NETworkS (COMSNETS)*. IEEE, 2022, pp. 779–783.
- [9] C. Li, C. Li, Y. Zhao, B. Zhang, and C. Li, "An efficient multi-model training algorithm for federated learning," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 1–6.
- [10] Z.-L. Chang, S. Hosseinalipour, M. Chiang, and C. G. Brinton, "Asynchronous multi-model federated learning over wireless networks: Theory, modeling, and optimization," *arXiv preprint arXiv:2305.13503*, 2023.
- [11] M. Siew, S. Arunasalam, Y. Ruan, Z. Zhu, L. Su, S. Ioannidis, E. Yeh, and C. Joe-Wong, "Fair training of multiple federated learning models on resource constrained network devices," in *Proceedings of the 22nd International Conference on Information Processing in Sensor Networks*, 2023, pp. 330–331.
- [12] C. Joe-Wong, S. Sen, T. Lan, and M. Chiang, "Multiresource allocation: Fairness-efficiency tradeoffs in a unifying framework," *IEEE/ACM Transactions on Networking*, vol. 21, no. 6, pp. 1785–1798, 2013.
- [13] E. Altman, K. Avrachenkov, and A. Garnaev, "Generalized  $\alpha$ -fair resource allocation in wireless networks," in *2008 47th IEEE Conference on Decision and Control*. IEEE, 2008, pp. 2414–2419.
- [14] E. Bamps, B. Escoffier, and S. Mladenovic, "Fair resource allocation over time," in *AAMAS 2018-17th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 766–773.
- [15] Y. Deng, F. Lyu, J. Ren, Y.-C. Chen, P. Yang, Y. Zhou, and Y. Zhang, "Fair: Quality-aware federated learning with precise user incentive and model aggregation," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.
- [16] T. Lan, D. Kao, M. Chiang, and A. Sabharwal, *An axiomatic theory of fairness in network resource allocation*. IEEE, 2010.
- [17] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," *arXiv preprint arXiv:1905.10497*, 2019.
- [18] Y. Zhang and Q. Yang, "A survey on multi-task learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 12, pp. 5586–5609, 2022.
- [19] Y. Ruan and C. Joe-Wong, "Fedsoft: Soft clustered federated learning with proximal local updating," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 7, 2022, pp. 8124–8131.
- [20] M. Hu, P. Zhou, Z. Yue, Z. Ling, Y. Huang, Y. Liu, and M. Chen, "Federoss: Towards accurate federated learning via multi-model cross aggregation," *arXiv preprint arXiv:2210.08285*, 2022.
- [21] W. Xia, T. Q. Quek, K. Guo, W. Wen, H. H. Yang, and H. Zhu, "Multi-armed bandit-based client scheduling for federated learning," *IEEE Transactions on Wireless Communications*, vol. 19, no. 11, pp. 7108–7123, 2020.
- [22] Y. J. Cho, J. Wang, and G. Joshi, "Towards understanding biased client selection in federated learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022, pp. 10351–10375.
- [23] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *2019 IEEE international conference on communications (ICC)*. IEEE, 2019, pp. 1–7.
- [24] J. Liu, J. Jia, B. Ma, C. Zhou, J. Zhou, Y. Zhou, H. Dai, and D. Dou, "Multi-job intelligent scheduling with cross-device federated learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 2, pp. 535–551, 2022.
- [25] M. N. Nguyen, N. H. Tran, Y. K. Tun, Z. Han, and C. S. Hong, "Toward multiple federated learning services resource sharing in mobile edge networks," *IEEE Transactions on Mobile Computing*, vol. 22, no. 1, pp. 541–555, 2021.
- [26] P. Poullie, T. Bocek, and B. Stiller, "A survey of the state-of-the-art in fair multi-resource allocations for data centers," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 169–183, 2017.

- [27] A. Lakhdari and A. Bouguettaya, "Fairness-aware crowdsourcing of iot energy services," in *Service-Oriented Computing: 19th International Conference, ICSOC 2021, Virtual Event, November 22–25, 2021, Proceedings 19*. Springer, 2021, pp. 351–367.
- [28] F. Basık, B. Gedik, H. Ferhatosmanoğlu, and K.-L. Wu, "Fair task allocation in crowdsourced delivery," *IEEE Transactions on Services Computing*, vol. 14, no. 4, pp. 1040–1053, 2018.
- [29] X. Gan, Y. Li, W. Wang, L. Fu, and X. Wang, "Social crowdsourcing to friends: An incentive mechanism for multi-resource sharing," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 3, pp. 795–808, 2017.
- [30] K. Donahue and J. Kleinberg, "Models of fairness in federated learning," *arXiv preprint arXiv:2112.00818*, 2021.
- [31] Y. Zhan, J. Zhang, Z. Hong, L. Wu, P. Li, and S. Guo, "A survey of incentive mechanism design for federated learning," *IEEE Transactions on Emerging Topics in Computing*, vol. 10, no. 2, pp. 1035–1044, 2021.
- [32] Y. Zhan, P. Li, Z. Qu, D. Zeng, and S. Guo, "A learning-based incentive mechanism for federated learning," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6360–6368, 2020.
- [33] J. Weng, J. Weng, H. Huang, C. Cai, and C. Wang, "Fedserving: A federated prediction serving framework based on incentive mechanism," in *IEEE INFOCOM 2021*. IEEE, 2021, pp. 1–10.
- [34] X. Wang, S. Zheng, and L. Duan, "Dynamic pricing for client recruitment in federated learning," *arXiv preprint arXiv:2203.03192*, 2022.
- [35] M. Tang and V. W. Wong, "An incentive mechanism for cross-silo federated learning: A public goods perspective," in *IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.
- [36] "Technical report." [Online]. Available: <http://tinyurl.com/MMFLpaper>
- [37] Y. Ruan, X. Zhang, and C. Joe-Wong, "How valuable is your data? optimizing client recruitment in federated learning," in *2021 19th International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt)*. IEEE, 2021, pp. 1–8.
- [38] Y. Singer, "Budget feasible mechanism design," *ACM SIGecom Exchanges*, vol. 12, no. 2, pp. 24–31, 2014.