An Efficiency-Boosting Client Selection Scheme for Federated Learning With Fairness Guarantee

Tiansheng Huang, Weiwei Lin, *Member, IEEE*, Wentai Wu, Ligang He,
Keqin Li, *Fellow, IEEE*, and Albert Y. Zomaya, *Fellow, IEEE*

Abstract—The issue of potential privacy leakage during centralized AI's model training has drawn intensive concern from the public. A Parallel and Distributed Computing (or PDC) scheme, termed Federated Learning (FL), has emerged as a new paradigm to cope with the privacy issue by allowing clients to perform model training locally, without the necessity to upload their personal sensitive data. In FL, the number of clients could be sufficiently large, but the bandwidth available for model distribution and re-upload is quite limited, making it sensible to only involve part of the volunteers to participate in the training process. The client selection policy is critical to an FL process in terms of training efficiency, the final model's quality as well as fairness. In this article, we will model the fairness guaranteed client selection as a Lyapunov optimization problem and then a $\mathbf{C^2MAB}$-based method is proposed for estimation of the model exchange time between each client and the server, based on which we design a fairness guaranteed algorithm termed RBCS-F for problem-solving. The regret of RBCS-F is strictly bounded by a finite constant, justifying its theoretical feasibility. Barring the theoretical results, more empirical data can be derived from our real training experiments on public datasets.

Index Terms—Client selection, contextual combinatorial multi-arm bandit, fairness scheduling, federated learning, lyapunov optimization

✦

1 INTRODUCTION

1.1 Background

FEDERATED Learning (FL) has been esteemed as one of the most promising solutions to the crisis known as isolated "data island". It helps break down the obstacles between parties or entities, allowing a greater extent of data sharing. All the entities being involved could benefit from such a new paradigm, in which model owners could build a more robust and comprehensive model with more data being accessible. Meanwhile, data owners might either receive substantial rewards or services that match their interests in return. More importantly, the privacy of the data owners would not risk being intruded since their raw data simply does not necessarily need to leave the local devices, as all the training is only performed locally.

- *Tiansheng Huang and Weiwei Lin are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, Guangdong 510641, China.*
  E-mail: cs_tianshenghuang@mail.scut.edu.cn, linww@scut.edu.cn.
- *Wentai Wu and Ligang He are with the Department of Computer Science, University of Warwick, CV4 7AL Coventry, United Kingdom.*
  E-mail: wentai.wu, Ligang.He@warwick.ac.uk.
- *Keqin Li is with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA. E-mail: lik@newpaltz.edu.*
- *Albert Y. Zomaya is with the School of Computer Science, The University of Sydney, Sydney, NSW 2006, Australia.*
  E-mail: albert.zomaya@sydney.edu.au.

1.2 Motivations

Within such a novel paradigm, new challenges co-exist with opportunities. Unlike the traditional model training process, not all the data within the system could be accessed over every round of training. Owing to the limited bandwidth and the dynamic status of the training clients, only a fraction of them could be picked to perform training on behalf of the model owner. From the perspective of a model owner, the selection decision in each round could have a profound impact on the model's training time, convergence speed, training stability, as well as the final achieved accuracy. Some studies in the literature have made iconic contributions to this problem. To illustrate, in [1], when making a selection, Nishio *et al.* concentrate on the evaluation of communication time, which accounts for a considerable portion of time for a training round. In another study [2], the authors consider more. They further take the energy consumption factor into consideration. Barring an intelligent decision on participant selection, an efficient bandwidth allocation scheme was also given by them. However, the current line of research evades two important factors. For one thing, both of them assume a pre-known local training time to the scheduler, which may not be realistic in all circumstances. For another, indicated by Theorem 2 in [2], devices with higher performance are more favored by their proposed methods. Indeed, always selecting the "fast" devices somehow boost the training process. But clients with low priority are simply being deprived of chances to participate at the same time, which we refer to it as an unfair selection among clients. In fact, such an extreme selection scheme might bring undesirable side effects by neutralizing some portions of data. Conceivably, with a smaller amount of data involved, data diversity can not be guaranteed,

thereby hurting the performance of model training to some extent. This motivates us to develop an algorithm that strikes a good balance between training efficiency and fairness. Also, the algorithm is supposed to be intelligent enough to predict the training time of the clients based on their reputation (or their historical performance), rather than assuming it to be known a priori.

1.3 Contributions

The main contributions of this paper are listed as follows:

1) We investigate the client selection in FL from the perspective of minimizing average model exchange time when subjecting to a relatively flexible long-term fairness guarantee, as well as a few rigid system constraints. At the same time, more factors, involving the clients' availability, unknown and stochastic training time, as well as the dynamic communication status, are taken into account.
2) Inspired by [3], we transform the original offline problem into an online Lyapunov optimization problem where the long-term guarantee of client participating rate is quantified using dynamic queues.
3) We build a Contextual Combinatorial Multi Arm Bandit (C^2MAB) model for estimation of the model exchange time of each client based on their contextual properties and historical performance (or their reputation).
4) A fairness guaranteed selection algorithm RBCS-F is proposed for efficiently resolving the proposed optimization problem in FL. Theoretical evaluation and real data-based experiments show that RBCS-F can ensure no violation in the long-term fairness constraint. Besides, the training efficiency has been significantly enhanced, while the final model accuracy remains close, in a comparison with random, i.e., the vanilla client selection scheme of FL.

To the best knowledge of the authors, this is the first trackable practice that combines Lyapunov optimization and C^2MAB for a long-term constrained online scheduling problem. Also, we shall remind the readers that the proposed combination does not confine to the application of our current proposed problem, but it has the potential to extend to a wider range of selection problems. (e.g., worker selection in crowdsensing, channel selection in the wireless network, etc.)

2 RELATED WORKS

In recent years, we are experiencing a great surge of Edge Intelligence (see in [4], [5], [6]). Numerous attempts have been made to combine AI techniques and edge, tapping the profound potential of the ubiquitous deployed edge devices. Among these, one of the most iconic studies could be neurosurgeon [7]. Its basic idea is to partition an intact DNN (Deep Neural Networks) into several smaller parts and disseminate them to the edge devices. Owing to a low latency between edge and users, inference speed could be significantly improved.

Besides, edge coordinated Federated Learning is another promising combination. Federated Learning [8], which allows data to be trained in local rather than being transmitted to the cloud, is now known as a more secure paradigm for AI's model training. We have witnessed the surge of some plausible applications of FL within these years (e.g., keyboard and emoji prediction in [9], [10], visual object detection in [11], etc). Despite the potential advantages as well as the promising applications of FL, the communication overhead between cloud and users renders as a bottleneck for it. A lengthy communication round during training might significantly degrade FL's training performance. Although more advance training schemes, such as federated distillation (FD, originally proposed in [12]), promise us a more desirable, reduced size information exchange between users and model aggregator, the latency between cloud and edge alone is inevitable. Such a defect could be better addressed by making edge the model's aggregator or at least an intermediate one (see in [13]). In this way, the data don't have to bear an outstanding communication length to the cloud. Another open problem of FL we would like to mention here is the client selection problem, originally proposed in [1] and followed by some related works (e.g., [2], [14], [15], [16]). Many of them see the problem from a communication perspective, focusing on building an efficient selection or bandwidth allocation scheme that helps shorten the communication length. In this paper, we will see the problem from a different angle, namely, to investigate how the fairness factor affects the training performance. We couldn't check out any specialized studies on this topic yet and we hope our research could bring some new insights in the field. Last but not least, we also want to note, FL itself is now far from its maturity, many important issues worth our study. Some of which might involve asynchronous or semi-asynchronous aggregation protocol [17], [18], incentive mechanism [19], [20] and security issues [21], etc. We look forward to more insightful and dedicated research into FL.

Now we would like to talk more about a classical problem, termed multi-arm bandit (MAB). In a classical MAB setting, arms are characterized by different unknown reward distribution. In each round of play, the player selects one of the arms from the possible options and gains a reward sampling from the selected arm's reward distribution. As there exists a tradeoff between exploration and exploitation for the player, how to maximize her obtained reward is the main concern. Several solutions, such as the well-known Upper Confidence Bound (UCB), Thompson Sampling (TS) could be applied to the problems. In addition, MAB has several variations. Those include combinatorial MAB, where players are allowed to select more than one arms in every round, contextual MAB [22], [23], where the reward of an arm follows a linear stochastic formulation, and a much newer one, contextual combinatorial MAB (C^2MAB) [24], [25], which is the combination of the above two. We found that C^2MAB could be well applied to the client selection problem in FL, as each client could be regarded as an arm and our task for each round is to choose a combination of which for participation, thus, in this paper, such a prototype will be used for our model establishment.

3 PRELIMINARY INTRODUCTION ON FL

In this paper, we consider an edge-coordinated federated learning system, in which edge is functioning as a model aggregator, and the clients (mostly mobile devices) are
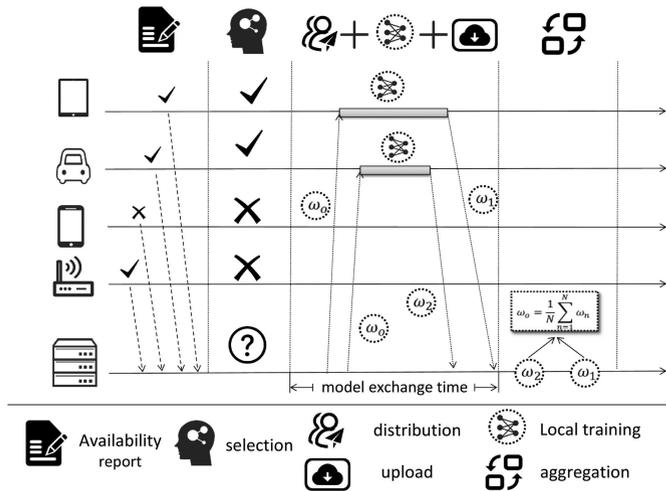
Fig. 1. Illustration of FL.

responsible for doing local training over their private data on behalf of the model's owner. We adopt in our system the most-accepted synchronous scheme for federated learning, which is characterized by training in iterations. For clearness, now we will explicitly explain the workflow of our synchronized scheme by giving four sequential stages of training, as follows:

1) At the very beginning of a new iteration, the clients first report their willingness to participate in the training as well as a few client-side information, which will be used for the client selection in the next stage.
2) In the second step, the scheduler conducts client selection to choose a portion of participants among the volunteers in light of the provided information.
3) Global model is distributed to the selected clients. After receiving the model, the clients conduct local training using their private data and update their local model. Once the training of all the selected clients is finished, the local model will be returned to the MEC server. The time span of this round is known as *model exchange time*.
4) The collected local models are aggregated by the server, substituting the original global model that once being distributed, and then it proceeds to step 1) to start a new iteration.

For a more vivid presentation of the training process, we refer the readers to Fig. 1.

4 PROBLEM FORMULATION

Our main concern focuses on the selection phase, in which the server makes a decision on the involved clients. Before our formal introduction of the selection problem, we first derive a high-level description of the content of this section. In the first sub-section, we formulate the client selection problem into an offline problem with a long-term fairness constraint. The formulated problem is simple in form but indeed unsolvable due to the time coupling effect as well as the unknown model exchange time persisting in the objective. To resolve the time coupling effect, we transform the

problem into an online mode using Lyapunov optimization technique, the online transformation of which gains us a fighting chance to derive an estimated model exchange time before each round scheduling, which might help resolve another obstacle (i.e., the unknown parameter in the objective function). Specifically, targeting the transformed online problem, a C^2MAB setting could come in handy for online learning of the exchange time, and being enlightened by which, we are able to further transform the problem into the ultimate form, which concludes the whole section.

Then we need to explain some key notations that are consistently used throughout the paper, among which, a set $\mathcal{T} \triangleq \{1, 2, \dots\}$, indexed by t, is used to capture the *federated rounds* (namely, the iterations in FL's model update process). The set $\mathcal{N} \triangleq \{1, 2, \dots N\}$ captures all the clients (each indexed by n) in the system. Besides, we assume that the maximum number of selected clients each round is fixed in advance to m. Another important notation is $\mathcal{S}_t$, which we use to capture the selected clients in round t and it serves as the representation of the selection policy that we aim to optimize.

4.1 Basic Assumption on System Model

4.1.1 Model Exchange Time

In a client selection problem, an important metric we shall evaluate is the long-term average model exchange time. We refer to the model exchange time as the time span between the instant the scheduler made the selection decision and that when all the re-upload models have been gathered. This model exchange time might involve time for model distribution, model training and model upload. Intuitively, a client selection scheme that is able to achieve a shorter span of each federated round is of interest, since a shorter period of each round explicitly marks shorter time for fix rounds of training. Recall that the server could step into the next phase (model aggregation) only after all the models have been gathered when adopting a synchronous federated training protocol. The time for model exchange is explicitly determined by the participated clients, or more precisely, by the one among them who spends the most time in training and model uploading. Mathematically, we have the following equation to capture the time span for a federated round:

$$f(\mathcal{S}_t, \boldsymbol{\tau}_t) = \max_{n \in \mathcal{S}_t} \{\tau_{t,n}\}, \qquad (1)$$

where we use a set $\mathcal{S}_t$ to capture the selected clients in round t. Besides, $\tau_{t,n}$ is used to represent the time span between the very beginning of model distribution and the instant when the model from client n being gathered. Here $\boldsymbol{\tau}_t \triangleq \{\tau_{t,n}\}_{n \in \mathcal{N}}$ in round t is unknown to the scheduler until the end of this round.

4.1.2 Long-Term Fairness Constraint

Another metric that might have a significant impact on FL's performance is fairness. Assume an ideal case that the server is fully aware of the exact model exchange time of each client for the incoming federated round. Then is it incontrovertibly optimized when always choosing the m-fastest clients, making the time span for each round of

training minimized? We must note, however, that the answer may not be such apparent. We acknowledge that the time span of each round could be somehow minimized by adopting such a greedy selection scheme, but we must argue that if we always choose the fastest clients, small chance could become available for their slower counterparts, implicitly implying that little contribution could be obtained from the slowers' local data. Very likely, along with the selection bias, the global model would suffer a degradation on its capability to generalize. In this regard, a greedy selection may not trivially be the best scheme, and fairness in selection is another factor that we need to take into account. To model such a critical fairness concern, we introduce a long-term fairness constraint, as follows:

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[x_{t,n}] \geq \beta \quad \forall n \in \mathcal{N}, \tag{2}$$

where β models the expected guaranteed chosen rate of clients. $x_{t,n}$ is used to indicate whether client n is involved in the federated round t or not. In other words, $x_{t,n} = 1$ for $n \in \mathcal{S}_t$; otherwise, $x_{t,n} = 0$. The constraint is set to make sure the long-term average chosen rate of every client at least greater than β, which somehow helps maintain some degrees of fairness for the system.

4.1.3 Availability of Clients

As we are investigating a client selection problem under a highly dynamic real-world system, it is unrealistic to assume clients are always ready to provide training services. In fact, clients are free to join and leave the loose "federation" at any time they want. With this consideration, we use an indicator function $I_{t,n}$ to capture the status of a client, indicating whether the client is willing to engage or not. Such information could be given by the availability report from the clients before scheduling. Formally, we introduce a strict constraint to prevent futile participation:

$$I_{t,n} = 1 \quad \forall n \in \mathcal{S}_t, \tag{3}$$

4.1.4 Selection Fraction

Recall that the maximum number of clients that could be selected is fixed to m in our setting. However, as the number of volunteers may not be able to reach m if the activated number is smaller than m, we have to use a "min" function to constraint the selection fraction, as follows:

$$|\mathcal{S}_t| = \min \left\{ m, \sum_{n \in \mathcal{N}} I_{t,n} \right\}, \tag{4}$$

where $|\mathcal{S}_t|$ means the number of elements in $\mathcal{S}_t$. Intuitively, in the case when the total number of availability could not overtake the maximum selection fraction, we simply involve all the active clients for the incoming round of training.

4.2 An Offline Long-Term Optimization Problem

Based on the above discussion, we are ready to introduce our client selection problem, as follows:

$$(P1): \min_{\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_\infty\}} \quad \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} f(\mathcal{S}_t, \boldsymbol{\tau}_t) \tag{5}$$
$$\text{s.t. } (2), (3), (4),$$

where $\mathcal{S}_t$ captures the selected clients in each round, which is our optimized target. Intuitively, our aim is to minimize the long-term model exchange time while subjecting to a "soft" long-term fairness constraint (2), which tolerates short-term violation, as well as two extra "hard" constraints (3), (4), which bear no compromise.

One could notice that $P1$ is a time-coupling scheduling problem, regarding the long-term objective and the fairness constraint in (2). But we note here that such an optimization problem is challenging or even impossible to be solved offline. There are mainly three concerns about this. First, random events, such as clients' availability, are not known to the scheduler until the very beginning of a particular round. This implies that an offline strategy, which is not given access to this particular information, can hardly guarantee the qualifications of constraints (3) and (4). Our second concern is derived from the time-coupling constraint (2), which is quite difficult for the offline solution to deal with. The final concern is that the information on model exchange time can only be observed after actually involving the clients in training. Nevertheless, the scheduler is supposed to make a scheduling decision before the real training process, when the actual model exchange time is unachievable. The lack of this crucial information precludes any feasible attempts to achieve an optimal offline solution. Therefore, for an alternative sub-optimal problem-solving, in the following section, we will elaborate on our transformation of the offline problem to a step-by-step online scheduling problem by Lyapunov optimization to cope with the first two proposed concern. Later, we will display our estimation of model exchange time based on clients' reputation, by which we leverage to deal with our third concern.

4.3 Problem Transformation Under Lyapunov Framework

In this sub-section, we first take advantage of Lyapunov optimization framework to transform the offline problem $P1$ to an online one.

First, we introduce a virtual queue for each client, whose backlog[1] is denoted by $Z_{t,n}$[2], to transform the long-term fairness constraint. Specifically, $Z_{t,n}$ evolves across the FL process complying the following rule:

$$Z_{t+1,n} = \left[Z_{t,n} + \beta - x_{t,n} \right]^{+}, \tag{6}$$

where β is the expected guaranteed selection rate in (2) and $[\ldots]^{+}$ is equivalent to $\max(\ldots, 0)$.

Now we present Theorem 1 to justify the rationale for this transformation.

---

1. We use the term "backlogs" and "queue length" interchangeably throughout the paper but actually, they share the same meaning.
2. The subscripts t and n here correspond to a federated round and a client, respectively. A similar form of subscript definition will be adopted throughout the paper.

Theorem 1. *Long-term time average constraint (2) holds if all the virtual queues (whose backlogs denoted by $Z_{t,n}$) remain* mean rate stable *across the FL process.*

Proof. According to the queue theory (see in Theorem 2.5, [26]), if all the virtual queues $Z_{t,n}$ remain *mean rate stable* across the FL process (or formally, $\lim_{T\to\infty} \mathbb{E}[Z_{T,n}]/T = 0$), the time average arrival rates of the queue will be smaller than the service rates, namely, we have:

$$\frac{1}{T} \lim_{T\to\infty} \sum_{t=1}^{T} \mathbb{E}[\beta - |\mathcal{S}_t|] \le 0. \qquad (7)$$

Through basic mathematics operations, we can reconstruct the above inequality into the form of (2) with ease. This completes the proof. □

Remark. Intuitively, the length of the queue will soar towards infinity if the long-term fairness constraint is violated, (i.e., when the real chosen rate could not match up with the expected guaranteed selection rate), which is formally justified by Theorem 1. To guarantee the fairness constraint, the queue has to remain *mean rate stable* and a qualified algorithm is supposed to achieve this goal. Apart from this conclusion, we shall note that the stabilized queue length could also reflect the degree of fairness. For example, if a client never being selected in the first few limited round, its corresponding queue length will soar to a positive value. After that, if its real selection rate basically flats with the expected guaranteed selection rate, its queue still remains *mean rate stable* and the queue length will slightly fluctuate over the same positive value. Intuitively, the bigger this value is, the unfairer the selection policy could be, as it demonstrates more violation of the fairness constraint in the initial stage. This conclusion could also be derived from the results in our experiments, which will be presented later.

With Theorem 1, now we have transformed the troublesome time-coupling constraint into the goal of ensuring the virtual queues mean rate stable across the FL process. To reach this end, a straightforward approach is to bound every increase of queues so that they could not grow to infinity. Under this motivation, we shall leverage Lyapunov optimization technique to bound the growth of virtual queues while simultaneously minimizing the objective in *P1*. First, we establish the quadratic Lyapunov function, with the following form:

$$\mathcal{L}(\Theta(t)) = \frac{1}{2} \sum_{n\in\mathcal{N}} Z_{t,n}^2, \qquad (8)$$

where $\Theta(t) \triangleq \{Z_{t,n}\}_{n\in\mathcal{N}}$ contains the backlogs of all the virtual queues.

Aiming at bounding the expected increase of $\mathcal{L}(\Theta(t))$ for one single round, we first formulate the *Lyapunov drift* to measure it, basically, we have:

$$\Delta(\Theta(t)) = \mathbb{E}[\mathcal{L}(\Theta(t+1)) - \mathcal{L}(\Theta(t))|\Theta(t)]. \qquad (9)$$

As the backlogs of queues $\Theta(t)$ can be known to the scheduler when being scheduled in an online manner, we take it as the condition in the Lyapunov drift. It is notable that the conditional expectation here is with respect to the availability of clients (which is a stochastic variable) as well as the possibly random selection policy. For ease of later interpretation, we let $\omega_t \triangleq \{I_{t,n}\}_{n\in\mathcal{N}}$ to capture the stochastic availability.

Recall that the objective of *P1* is to minimize the model exchange time while satisfying the given constraints. This motivates us to combine the objective function into the drift function. Formally, we term such a combination as *drift-plus-cost* function, with the following form:

$$\Delta(\Theta(t)) + V\mathbb{E}[f(\mathcal{S}_t, \boldsymbol{\tau}_t)|\Theta(t)], \qquad (10)$$

where $V \ge 0$ is a penalty factor set for the purpose of balancing the tradeoff between minimizing the objective and satisfying the fairness constraint. Such a parameter is crucial for the algorithm's performance and we will conduct a specific analysis to it in the next section. Note that the conditioned expectation being taken here is also with respect to stochastic events $\omega(t)$ and the possibly random policy as well. Now we are going to introduce a potential upper bound for the drift-plus-cost function. We show the result by Theorem 2.

Theorem 2. *Conditioning on the queues' backlogs $\Theta(t)$, the drift-plus-cost function for our system model could be bounded into the following form, where $\Gamma = N(1 + \beta^2)/2$ is a constant.*

$$\begin{aligned} &\Delta(\Theta(t)) + V\mathbb{E}[f(\mathcal{S}_t, \boldsymbol{\tau}_t)|\Theta(t)] \\ &\le \Gamma + \sum_{n\in\mathcal{N}} Z_{t,n}\mathbb{E}[\beta - x_{t,n}|\Theta(t)] + V\mathbb{E}[f(\mathcal{S}_t, \boldsymbol{\tau}_t)|\Theta(t))]. \end{aligned}$$

$$(11)$$

Proof. The complete proof is given in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2020.3040887. □

Intuitively, if we minimize the Right Hand Side (R.H.S) of (11), the fairness virtual queues could be somehow maintained stable, while the objective function is also being minimized. Now shall introduce our step-by-step online scheduling problem by giving *P2*:

$$(P2): \min_{\boldsymbol{x}_t} \quad \Gamma + \sum_{n\in\mathcal{N}} Z_{t,n}(\beta - x_{t,n}) + V\dot{f}(\boldsymbol{x}_t, \boldsymbol{\tau}_t)$$

$$s.t. \quad \sum_{n\in\mathcal{N}} x_{t,n} = \min\left\{m, \sum_{n\in\mathcal{N}} I_{t,n}\right\} \qquad (12)$$

$$x_{t,n} \le I_{t,n}$$

$$x_{t,n} \in \{0,1\},$$

we first have to make it clear that we use $\boldsymbol{x}_t$ to substitute all the $\mathcal{S}_t$ in *P1*, making it a clearer form. Here $\dot{f}(\boldsymbol{x}_t, \boldsymbol{\tau}_t) = \max_{n\in\mathcal{N}}\{x_{t,n}\tau_{t,n}\}$ is an equivalent form to $f(\mathcal{S}_t, \boldsymbol{\tau}_t)$. While solving *P2* on every round, the R.H.S of (11) can be minimized. The rationale behind is quite evident. As we have done the minimization under every round (alternatively, under every ω_t, since ω_t is an independent sampling for each round), then the expectation with respect to ω_t is also being minimized. Note here that ω_t is indeed observable for an online algorithm since an online algorithm makes scheduling after the stage of availability report, making it accessible to this particular information.

For briefness, we eliminate all the constants (i.e., Γ, $Z_{t,n}\beta$) in the objective of $P2$ and transform it to $P3$:

$$(P3): \min_{\mathbf{x}_t} \quad V \max_{n \in \mathcal{N}} \{x_{t,n} \tau_{t,n}\} - \sum_{n \in \mathcal{N}} Z_{t,n} x_{t,n}$$

$$s.t. \quad \sum_{n \in \mathcal{N}} x_{t,n} = \min \left\{ m, \sum_{n \in \mathcal{N}} I_{t,n} \right\}$$

$$x_{t,n} \leq I_{t,n} \tag{13}$$

$$x_{t,n} \in \{0, 1\}.$$

But note that such a problem remains unsolvable yet since the real model exchange time of all the clients (or $\tau_{t,n}$) is not known to us before real scheduling. In the next subsection, we will present a C^2MAB estimation to conquer such a barrier.

4.4 Estimation of Model Exchange Time With C^2MAB

4.4.1 Background Knowledge on C^2MAB and UCB

Each round selection in a Contextual Combinatorial Multi Arm Bandit (C^2MAB) is characterized by a tuple $(\mathcal{N}, \mathcal{S}_t, \{\boldsymbol{\theta}_n^*\}_{n \in \mathcal{N}}, \{\boldsymbol{c}_{t,n}\}_{n \in \mathcal{N}}, \{\epsilon_{t,n}\}_{n \in \mathcal{N}}, f(\cdot))$, in which $\mathcal{N}$ represents the arm set and $\mathcal{S}_t$ is another set that catpures all the possible combination of arms. $\boldsymbol{c}_{t,n}$ and $\boldsymbol{\theta}_n^*$ represents the contextual vector and coefficient vector respectively, among which, $\boldsymbol{c}_{t,n}$ is known before each round scheduling but dynamic between rounds, while $\boldsymbol{\theta}_n^*$ is unknown but stationary. After each round of scheduling, a combination of arms (often being called as a super arm) $S_t \subset \mathcal{S}_t$ is put into play. Then loss drawn from each selected arm, formulated by $l_{t,n} = \boldsymbol{c}_{t,n}^\top \boldsymbol{\theta}_n^* + \epsilon_{t,n}, n \in S_t$ is revealed to the scheduler, and meanwhile, a collective loss $f(\{r_{n,t}\}_{n \in S_t})$ is imposed. Our ultimate aim in the C^2MAB setting is to minimize the expected cumulative penalty $\frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[f(\cdot)]$ as far as possible by a careful selection on S_t.

Now we shall give a high-level description of a plausible solution for C^2MAB, i.e., a UCB algorithm. The UCB algorithm takes the upper confidence bound as the optimistic estimation of the expected loss in each round. As the historical data accumulated, (i.e., $l_{t,n}$ in the previous rounds), the bound could be narrowed and eventually converges to the real value, and thereby gaining more precision for our scheduling. By this means, the expected cumulative penalty could be minimized to the full extent with the increase of rounds of play.

4.4.2 Application

Recall that the information of model exchange time, or at least an estimated one, is supposed to be fetched before real client selection. One can take advantage of a MAB based technique to predict the model exchange time for all clients based on their historical performance (or to say, their reputation). In particular, each client can be regarded as an arm[3]. in a bandit setting and a combination of them (i.e., a super arm) is put into training, after which, the model exchange time for the selected arm, namely, $\{\tau_{t,n}\}_{n \in \mathcal{S}_t}$ can be observed by the scheduler.

Normally, the model exchange time is associated with the client's computation capacity, running status as well as the bandwidth allocation for the model update. In this regard, we consider introducing linear contextual bandit into our estimation. Formally, we let $\boldsymbol{c}_{t,n} \triangleq [1/\mu_{t,n}, s_{t,n}, M/B_{t,n}]^\top$ denote the contextual feature vectors that are collected by the scheduler before the scheduling phase. More explicitly, $\mu_{t,n}$ is the ratio of available computation capacity of client n over round t. We can simply comprehend $\mu_{t,n}$ as the available CPU ratio of the client.[4] A binary indicator $s_{t,n}$ indicates if client n has participated in training in the last round. M is the size of the model's parameters (measured by bit) and $B_{t,n}$ indicates the allocated bandwidth. Barring the available computation capacity of clients (i.e., $\mu_{t,n}$), which have to be proactively reported by the clients, all the other information could be fetched by the servers with ease. Therefore, here we can just comprehend the contextual feature $\boldsymbol{c}_{t,n}$ as some prior information known by us before we do the scheduling. Given the contextual features, we assume that the sampling value of $\tau_{t,n}$ complies with the following equation:

$$\tau_{t,n} = \boldsymbol{c}_{t,n}^\top \boldsymbol{\theta}_n^* + \epsilon_{t,n}, \tag{14}$$

where $\boldsymbol{\theta}_n^* \triangleq [\tau_n^b, \tau_n^s, 1/\eta]^\top$ captures the static coefficient factors that are presumed to be unknown to the scheduler as they are hard to be detected by the server or even by the clients themselves. More explicitly, τ_n^b is the local training time for 100 percent computation capacity. Multiplying it with the first element in $\boldsymbol{c}_{t,n}$, we get the approximated local training time under the computation capacity provided by clients. τ_n^s denotes the cold start time, multiplying which with the second element $s_{t,n}$ in contexts yields the real data preparation time. This formulation is derived from the fact that clients who did not undertake the previous round of training need to spend extra time for data preparation, say, loading the data into memory. Likewise, we let $\eta \triangleq \log(1 + \text{SNR})$ and multiplying which with $B_{t,n}$ yields the Shannon formula that we use to calculate the uploading data rate. Here SNR is an abbreviation of Signal-to-Noise Ratio, which is associated with the client profile (e.g., transmission power and channel condition). In this regard, $M/(B_{t,n}\eta)$ can fully represent the model uploading time for client n. In light of our formulation, $\boldsymbol{c}_{t,n}^\top \boldsymbol{\theta}_n^*$ yields the approximation of the expected model exchange time.

In addition, acknowledging some deviation, we admit a noise factor $\epsilon_{t,n}$ in our estimation, which is assumed to be a zero-mean random variable, conditionally sampling from an unknown distribution with left-bounded support, i.e., $\text{Supp}(\epsilon_{t,n}|\boldsymbol{c}_{t,n}^\top) = (a, b)$ where $a > -\boldsymbol{c}_{t,n}^\top \boldsymbol{\theta}_n^*$ and b is arbitrary. This assumption is made to ensure that $\tau_{t,n}$ must be always positive. Also, we have to make sure that $\epsilon_{t,n}$ is conditionally R-sub-Gaussian where $R \geq 0$ is a fixed constant. Formally, we need:

$$\forall \Lambda \in \mathbb{R} \quad \mathbb{E}\left[e^{\Lambda \epsilon_{t,n}} \mid \boldsymbol{c}_{1:t,n}, \epsilon_{1:t-1,n}\right] \leq \exp\left(\frac{\Lambda^2 R^2}{2}\right). \tag{15}$$

---

3. We use an arm to represent a specific client in our later analysis

4. Note that $\mu_{t,n}$ could exceed 100 percent since a client could have more than 1 CPUs, say, $\mu_{t,n} = 200\%$ when 2 CPUs are free.

This assumption is necessary for the regret analysis of a linear bandit, which is also adopted by [23]. Though we admit some loss of generality for the noise assumption, we argue that a great number of distribution families in nature corresponds to R-sub-Gaussian (e.g., any distributions with zero mean bounded support, zero-mean Gaussian distribution, etc), so the assumption would not compromise the objectivity of this paper.

Now we let $\tau_{t,n}^* = \mathbb{E}[\tau_{t,n}] = \boldsymbol{c}_{t,n}^\top \boldsymbol{\theta}_n^*$. If $\tau_{t,n}^*$ is clearly known to us, we can safely substitute $\tau_{t,n}$ in $P3$ with it. Recall that $\boldsymbol{\theta}_n^*$ is an inherent feature of each arm (or client) that is supposed to be static, unchangeable over time. With this assumption, although the scheduler has no access to the real value of $\boldsymbol{\theta}_n^*$, which creates a barrier in the calculation of $\tau_{t,n}^*$, this value can be predicted using the historical information (or the reputation of an arm). For such a linear formulation, ridge regression could suit well. Now we let $(\mathbf{D}_{t,n}, \mathbf{y}_{t,n})$ to represent p pieces of client n's historical performance (i.e., the previous model exchange time and the contexts) that are obtained before round t. Formally, we have:

$$\mathbf{D}_{t,n} = \begin{bmatrix} \mathbf{c}_n^{(1)} \\ \vdots \\ \mathbf{c}_n^{(p)} \end{bmatrix}_{m \times 3} \quad \mathbf{y}_{t,n} = \begin{pmatrix} \tau_n^{(1)} \\ \vdots \\ \tau_n^{(p)} \end{pmatrix} \quad (16)$$

where $\mathbf{c}_n^{(p)}$ and $\tau_n^{(p)}$ respectively represent the context and the real model exchange time of the pth play of the arm n. With ridge regression, we can empirically estimate $\boldsymbol{\theta}_n^*$ with $\hat{\boldsymbol{\theta}}_{t,n}$:

$$\hat{\boldsymbol{\theta}}_{t,n} = \left(\mathbf{D}_{t,n}^\top \mathbf{D}_{t,n} + \lambda \mathbf{I}_3 \right)^{-1} \mathbf{D}_n^\top \mathbf{y}_{t,n}. \quad (17)$$

For ease of algorithm's design, we then transform $\hat{\boldsymbol{\theta}}_{t,n}$ into an equivalent form, as follows:

$$\hat{\boldsymbol{\theta}}_{t,n} = \mathbf{H}_{t-1,n}^{-1} \mathbf{b}_{t-1,n}, \quad (18)$$

where $\mathbf{H}_{T,n} = \mathbf{H} + \sum_{t=1}^{T} x_{t,n} \mathbf{c}_{t,n} \mathbf{c}_{t,n}^\top$ and $\mathbf{b}_{T,n} = \sum_{t=1}^{T} x_{t,n} \tau_{t,n} \mathbf{c}_{t,n}$. Among which, $\mathbf{H} = \lambda \mathbf{I}$.

As we are going to take advantage of the UCB algorithm we previously discussed as our solution, we resort to $\bar{\tau}_{t,n}$ as the optimistic estimation of $\tau_{t,n}$, which has the following form:

$$\bar{\tau}_{t,n} \triangleq \max\left\{ \mathbf{c}_{t,n}^\top \hat{\boldsymbol{\theta}}_{t,n} - \alpha_t \sqrt{\mathbf{c}_{t,n}^\top \mathbf{H}_{t-1,n}^{-1} \mathbf{c}_{t,n}}, 0 \right\}, \quad (19)$$

where α_t is an exploration parameter.

Now we show in Lemma 1 the validity of the given confidence bound (i.e., to demonstrate that the real expected exchange time does not deviate much from the confidence bound with a high probability).

Lemma 1. *If we set* $\alpha_t = R\sqrt{3 \log\left(\frac{1+tL^2/\lambda}{\delta}\right)} + \lambda^{1/2}S$, *with probability at least* $1 - \delta$, *we have*

$$0 \leq \tau_{t,n}^* - \bar{\tau}_{t,n} \leq 2\alpha_t \|\mathbf{c}_{t,n}\|_{\mathbf{H}_{t-1,n}^{-1}}, \quad (20)$$

for any round $t \geq 1$ *and any arm* $n \in \mathcal{N}$

Proof. The complete proof is given in Appendix B, available in the online supplemental material. □

We first note here that Lemma 1 will be used in our analysis of regret bound, which will be shown in the next section.

As we have decided $\bar{\tau}_{t,n}$ as our estimation of $\tau_{t,n}$, we now transfer $P3$ to the ultimate form, shown in the following:

$$(P4) : \min_{\mathbf{x}_t} \quad V \max_{n \in \mathcal{N}}\{x_{t,n} \bar{\tau}_{t,n}\} - \sum_{n \in \mathcal{N}} Z_{t,n} x_{t,n}$$

$$s.t. \quad \sum_{n \in \mathcal{N}} x_{t,n} = \min\left\{ m, \sum_{n \in \mathcal{N}} I_{t,n} \right\}$$

$$x_{t,n} \leq I_{t,n} \quad (21)$$

$$x_{t,n} \in \{0, 1\}.$$

Then transformed problem is an Integer Linear Programming (ILP) problem, which is indeed theoretically solvable and for which we design a divide-and-conquer-based algorithm for an efficient settlement, shown in the coming section.

5 ALGORITHMS AND ANALYSIS

In this section, we first present the detail of our proposed algorithm, and then some related analysis is given.

5.1 Algorithms Design

Noticeably, the first term on the objective function of $P4$ has only finite possible values, so we can simply iterate these values and transform them into the constraint in the sub-problems. By this means, we divide the problem into a few smaller-scale sub-problems, which are easier to conquer. Formally, the sub-problem after division is shown in the following:

$$(P4 - SUB) : \min_{\mathbf{x}_t} \quad - \sum_{n \in \mathcal{N}} Z_{t,n} x_{t,n}$$

$$s.t. \quad \sum_{n \in \mathcal{N}} x_{t,n} = \min\left\{ m, \sum_{n \in \mathcal{N}} I_{t,n} \right\}$$

$$x_{t,n} \bar{\tau}_{t,n} \leq \bar{\tau}_{max}$$

$$x_{t,n} \leq I_{t,n}$$

$$x_{t,n} \in \{0, 1\}, \quad (22)$$

where $\bar{\tau}_{max}$ is one of the fixed value among the possible values of the first term in $P4$. $P4$-SUB is much easier to conquer. First we only need to filter those qualified clients with a smaller or equal $\bar{\tau}_{t,n}$ to $\bar{\tau}_{max}$, and with an active status (or to say $I_{t,n} = 1$). Trivially, the sub-problem can be solved by finding $k = \min\{m, \sum_{n \in \mathcal{N}} I_{t,n}\}$ clients with the biggest $Z_{t,n}$ among the qualified clients. After the divide-and-conquer process, we only need to compare all the objectives obtained from the sub-problems and select the minimum one as our final achieved solution. The detail of the above process can be found in Algorithm 1, which could at least reach a computation complexity of $\mathcal{O}(N^2)$.

With Algorithm 1 introduced, now we shall discuss our proposed solution for fairness-aware FL, termed Reputation Based Client Selection with Fairness (RBCS-F), shown in Algorithm 2. The working procedure of RBCS-F is quite intuitive. The algorithm starts with initialization of some parameters in the first three lines, and then begins to start iterative federated learning. In every iteration, the scheduler first observes the contexts and the availability of the arms (i.e., FL clients), then estimates the model exchange time with Eqs. (18) and (19) using historical information. Taking

advantage of the observed context, availability as well as the estimation, the selection scheme for this round could be fetched by Algorithm 1. After the decision, the model would be distributed to the selected clients and gathered after local training. Before the end of a round, the algorithm records the exchange time of the selected clients and update the associated parameters, as shown in lines 14-16.

Algorithm 1. Divide-and-Conquer Solution for *P4*

Input:
  The estimated time for model exchange; $\{\bar{\tau}_{t,n}\}_{n \in \mathcal{N}}$
  The expected number of chosen arms; m
  Indicator function of arms' availability; $\{I_{t,n}\}_{n \in \mathcal{N}}$
  Length of virtual queue; $\{Z_{t,n}\}_{n \in \mathcal{N}}$
Output:
  The solution for *P4* in round t; $\{x_{t,n}\}_{n \in \mathcal{N}}$
 1: Set $Z_t^* = \{Z_{t,n}\}_{I_{t,n}=1}$
 2: Use $\mathcal{A}_t$ to store arms with an descending order of Z_t^*
 3: Use $\mathcal{N}_t^+$ to store all the n that satisfies $I_{t,n} = 1$
 4: Set $k = \min\{m, \sum_{n \in \mathcal{N}} I_{t,n}\}$ // # of clients to be picked
 5: **for** $n_{max} \in \mathcal{N}_t^+$ **do**
 6:     Initialize an empty set $\mathcal{S}_{n_{max}}$
 7:     **for** $n \in \mathcal{A}_t$ **do**
 8:         **if** $\bar{\tau}_{t,n} \le \bar{\tau}_{t,n_{max}}$ **then**
 9:             Push n into $\mathcal{S}_{n_{max}}$
10:         **end if**
11:         **if** $length(\mathcal{S}_{n_{max}}) == k$ **then**
12:             Calculate the objective of *P4* as $F_{n_{max}}$ based on $\mathcal{S}_{n_{max}}$
13:             Break the first loop
14:         **end if**
15:     **end for**
16: **end for**
17: Set n^* the index of minimum $F_{n_{max}}$ among those being calculated in line 12.
18: Return $\{x_{t,n}\}$ that represented by $\mathcal{S}_{n^*}$

5.2 Theoretical Analysis

5.2.1 Regret and Fairness Guarantee

In an MAB model, regret is a key performance metric that measures the performance gap between a given policy and the optimal policy. Therefore, for ease of analysis, we first define the time average regret of RBCS-F.

Definition 1. *Time average regret of RBCS-F is defined as:*

$$R(T) \triangleq \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[f(\mathcal{S}_t, \boldsymbol{\tau}_t) - f(\mathcal{S}_t^*, \boldsymbol{\tau}_t)\right], \tag{23}$$

where we leverage $\mathcal{S}_t^*$ to represent the decision made by the optimal policy while $\mathcal{S}_t$ captures RBCS-F's decision.

To proceed, we show a strict upper bound on time average regret of RBCS-F in Theorem 3.

Theorem 3. *Given any control parameter V, with probability at least $(1 - \delta)^2$, the time average regret achieved by RBCS-F is upper bounded by:*

$$R(T) \le \frac{N(1+\beta^2)}{2V} + \zeta_T \sqrt{\frac{6\log(1+TL^2/3\lambda)}{T}}, \tag{24}$$

where S and L are both positive finite constants satisfying $\|\theta_n^*\|_2 \le S$ and $\|\mathbf{c}_{t,n}\|_2 \le L$ for all $t \ge 1$ and $n \in \mathcal{N}$. And:

$$\zeta_T = \max\{K, 1\} \cdot \max\left\{2R\sqrt{3\log\left(\frac{1+TL^2/\lambda}{\delta}\right)} + \lambda^{1/2}S, 1\right\}$$

where K is a constant value.

Proof. The complete proof is given in Appendix C, available in the online supplemental material. □

Now we give another theorem to ensure that the long-term fairness constraint would not be violated.

Theorem 4. *For RBCS-F, the fairness vitual queues are all mean rate stable in any setting of V, thus the time average fairness is being guaranteed.*

Proof. The complete proof is given in Appendix D, available in the online supplemental material. □

5.2.2 Impact of V

In light of Theorem 3, it seems quite reasonable for us to set the penalty factor V as large as possible so as to eliminate the first term in the regret upper bound. Such an extreme setting seems even more attractive regarding the fact that the long-term fairness constraint holds under any setting of V, which is justified by Theorem 4. Although a large value of V could indeed bring us a more satisfying long-term model exchange time while satisfying the long-term fairness constraint, we must claim here that the fairness factor is *not* impervious to the setting of V. Note that our long-term fairness constraint is built on the premise that the training rounds are infinite, but this may not be true in real training. With a larger V, the fairness queue will have a slower rate to converge, indicating that fairness could not be well guaranteed before convergence. When the training rounds are finite, the number of rounds that need to undergo before convergence could compromise some degrees of fairness. Such an analysis could be verified by our experiment results that we are now going to display.

6 EXPERIMENTS

In this section, we present the detail of our experiments. In the first sub-section, we would explain the general setting of our simulation environment and evaluate the numerical performance of our proposed solutions. The numerical evaluation results could well explain the relationship between the penalty factor (V), fairness (reflected by the queue length), and efficiency guarantee (the time span of a federated round). Then we will move on to the evaluation of the real training of two iconic public datasets, CIFAR-10 and fashion-MNIST, both of which are evaluated under different settings of non-iid extent. The real-data experiment will show how our proposed RBCS-F impacts the training efficiency and final model performance (i.e., accuracy).

6.1 Numerical Simulation

6.1.1 Simulation Setting

In our simulation, we assume the model exchange time conforms to the linear formulation as shown in Eq. (14). To simulate a heterogeneous system with clients of different computation and communication capacity, we equally

TABLE 1
Inherent Setting of Arms (or Clients)

client	τ_n^b	τ_n^s	η
class		(cold start time)	$\log(1 + SNR)$
1	1s	1s	$\log(1 + 1000)$
2	2s	1s	$\log(1 + 100)$
3	3s	1s	$\log(1 + 10)$
4	4s	1s	$\log(1 + 1)$

divide the total number of 40 clients into 4 classes and accordingly endow disparate abilities to them. For clearness, one can check Table 1 for the inherent training setting of different classes of clients.

Algorithm 2. Reputation Based Client Selection with Fairness (RBCS-F)

Input:

    The expected number of involved clients each round; m
    Exploration parameter; $\alpha_0, \alpha_1, \dots$
    The set of clients; $\mathcal{N}$, Parameter for ridge regression; λ
    The guaranteed participating rate; β
    Parameter for objective balance; V

Output:

    The control policy $\pi = \{x_{t,n}\}_{n \in \mathcal{N}, t=0,1,\dots}$

1: **for** $n \in \mathcal{N}$ **do**
2:    Initialize $\mathbf{H}_{0,n} \leftarrow \lambda \mathbf{I}_{3 \times 3}$, $\mathbf{b}_{0,n} \leftarrow \mathbf{0}_3^\top$, $Z_{0,n} \leftarrow 0$
3: **end for**
4: **for** $t = 1, 2 \dots$ **do**
5:    Observe current contexts $\{\mathbf{c}_{t,n}\}$ and arms availability $\{I_{t,n}\}$
6:    **for** $n \in \mathcal{N}$ **do**
7:        $\hat{\theta}_{t,n} \leftarrow \mathbf{H}_{t-1,n}^{-1} \mathbf{b}_{t-1,n}$
8:        $\hat{\tau}_{t,n} \leftarrow \mathbf{c}_{t,n}^\top \hat{\theta}_{t,n}$
9:        $\bar{\tau}_{t,n} \leftarrow \hat{\tau}_{t,n} - \alpha_t \sqrt{\mathbf{c}_{t,n}^\top \mathbf{H}_{t-1,n}^{-1} \mathbf{c}_{t,n}}$
10:   **end for**
11:   // Execute Algorithm 1 for a decision
        $\{x_{t,n}\} \leftarrow$ Algorithm 1$(\{\bar{\tau}_{t,n}\}, m, \{I_{t,n}\}, \{Z_{t,n}\})$
12:   Distribute model to the selected clients and observe their model exchange time; $\{\tau_{t,n}\}$
13:   **for** $n \in \mathcal{N}$ **do**
14:      Update $Z_{t,n}$ according to (6)
15:      $\mathbf{H}_{t,n} \leftarrow \mathbf{H}_{t-1,n} + x_{t,n} \mathbf{c}_{t,n} \mathbf{c}_{t,n}^\top$
16:      $\mathbf{b}_{t,n} \leftarrow \mathbf{b}_{t-1,n} + x_{t,n} \tau_{t,n} \mathbf{c}_{t,n}$
17:   **end for**
18: **end for**

For the context generation (in order to simulate the per-round status of clients), we assume the allocated bandwidth of all clients is sampling from a uniform distribution between [2,4] MHz and the model size M is fixed to 20 Mb. Likewise, the available computation capacity of all clients is also sampling from the same uniform distribution within [50%, 200%]. The indicator $s_{t,n}$ is set according to the training decision in the last round. In addition, for the noise in our linear formulation, we draw ϵ from a conditional uniform distribution within $(-\mathbf{c}_{t,n}^\top \theta_n^*, \mathbf{c}_{t,n}^\top \theta_n^*)$. The availability of clients follows the same Bernoulli distribution with parameter 0.8, and the setting of other algorithm related parameters could be found in Table 2. In our simulation, we mainly compare RBCS-F with two baseline selection methods that are commonly used in the field, i.e., random and FedCS [1]. Note that we

TABLE 2
Parameters Setting

notation	meaning	value
β	guaranteed participating rate	0.15
m	maximum selected clients	8
λ	parmeters for ridge regression	1
α_t	exploration factor	0.1
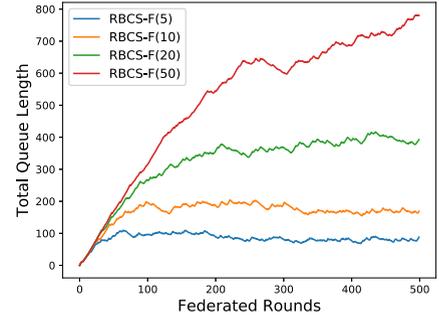
Fig. 2. The impact of V on the convergence of queues.

have made an adaption to FedCS in order to accommodate it to our context, but the basic idea is the same as the vanilla one, which is to select as much as clients within a fixed deadline. More concretely, we allow FedCS to have full access to both the contextual features and the static coefficient factor. With the additional information, its strategy is to select all the clients that possess an expected training time (i.e., $\mathbf{c}_{t,n}^\top \theta_n^*$) that shorter than the pre-set deadline.

6.1.2 Numerical Performance Evaluation

In our first evaluation, we show the variation of queue status for RBCS-F under different values of penalty factor V. As shown in Fig. 2, where RBCS-F(x) is abbreviated for RBCS-F with a penalty of $V = x$, it is interesting to see that all the curves with different settings of V flatten after going through a number of scheduling rounds. This phenomenon can justify our conclusion of the mean rate stability of the queues, which indicates that they could not grow to infinity and break our fairness constraint. Another observation we can derive here is that the curve with a higher penalty factor (i.e., V) seems to have a slower convergence speed and a higher convergence value. This implies that a large value of V might sacrifice a few fairness before its convergence, although it does conform to the long-term fairness constraint. Such an observation is consilient with our explanation given in the remark below Theorem 1 and our theoretical analysis in the last section.

Now we take a look at the evolution of training time across scheduling rounds. In Fig. 3, we depict the time consumption of our proposed RBCS-F with different V, and that of the random strategy and FedCS(3) [5]. As depicted, RBCS-F seems to have a satisfying enhancement in reducing the training time, compared with the random scheme, and of the same number of federated rounds, RBCS-F with a

---

5. For FedCS(3), we set its deadline (one of its key parameter) to 3s. The specific setting allows us to make the number of its selection clients approximates to 8, which is exactly the selection number of other strategies (see our setting in Table 2).
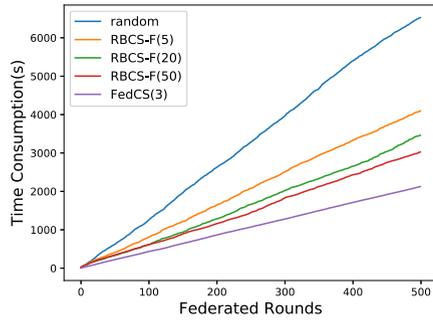
Fig. 3. Training time of different client-selection strategies.

Fig. 4. Pull record of arms (or clients) under different client-selection strategies.

higher V boasts a shorter time consumption. In addition, it is interesting to see that there is a performance gap between RBCS-F and FedCS(3). We note that this gap is inevitable due to our introduction of the fairness factor and the cost of online learning, but the bound itself is well-defined by our analysis of the regret.

The variance in training time of different schemes could be alternatively explained by looking at Fig. 4. This figure depicts the pull number of different arms (or chosen times of FL clients) after going through 500 rounds of decision, in which the clients are sorted in ascending order over their pull number. The brighter color indicates a heavier pull (or more times being selected) on the corresponding arm. From Fig. 4, we notice that the pull number of clients could vary dramatically when V is set to a high value and the unbalanced selection is more intense for FedCS(3). By contrast, the scheme that is known to be fairer (e.g., random or RBCF-F with low penalty) boasts an even distribution on the pull number, based on which we can explain why the training time of RBCS-F would escalate with a fairer selection. Clearly, the selection scheme that evenly chooses the clients shall never match up with those always choosing the fastest ones. However, is it the faster the better? Does fairness matter in real training? Now we are going to explore the answers with our real training on two public datasets.

6.2 Training on Public Dataset

6.2.1 Setup

We set up federated environment with *PyTorch* (version: 1.6.0) and all the computation is conducted using a high-performance workstation (Dell PowerEdge T630 with 2x GTX 1080Ti). We have prepared two tasks for an evaluation purpose. To be specific, we use two different Convolutional Neural Network (CNN) models to predict the classifying results from two datasets, fashion-MNIST, and CIFAR-10. For fashion-MNIST, we adopt a CNN with two 5x5 convolution layers (the first with 20 channels, the second with 50, each followed with 2x2 max pooling), a fully-connected layer with 500 units and ReLu activation, and finally a softmax output layer. For CIFAR-10, which is known to be a harder task, we use another much heavier CNN model with two 5x5 convolution layers (each with 64 channels), also followed with 2x2 max pooling, two fully connected layers with respective 384 and 192 units, and finally a softmax output layer.

In addition to the general iid setting, we also explore the training performance on a non-iid one. Here we adopt the same approach as in [27] to synthesize non-identical client
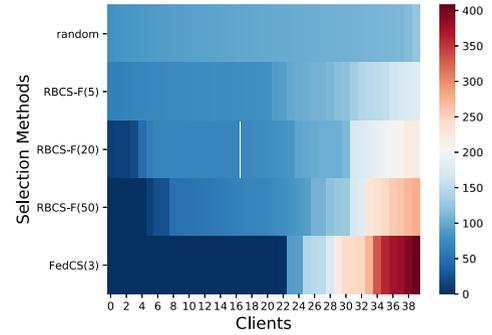
data. More specifically, we uniformly sample $q_i \times 500$ items from each of the classifying class, where $\boldsymbol{q} \triangleq (q_1, q_2, \ldots, q_i)$ is drawn from a Dirichlet distribution, i.e., $\boldsymbol{q} \sim \mathrm{Dir}(\gamma_1 \boldsymbol{p})$. Here $\boldsymbol{p}$ is an all-1 10-dimension vector [6] and γ_1 is a *concentration* parameter controlling the extent of identicalness among clients, say, with $\gamma_1 \to 0$ each client holds only one class chosen at random (i.e., high degree of non-iid), conversely, all clients have identical access to all classes (i.e., approximates to iid) if $\gamma_1 \to \infty$.

6.2.2 Impact of Fairness

In order to quantify the fairness factor and investigate how the factor affects the model accuracy as well as the training efficiency, we thereby introduce γ_2 to indicate the extent of fairness. Analogically to how we quantify the non-iid extent, we draw $\boldsymbol{q}$ from a Dirichlet distribution, i.e., $\boldsymbol{q}' \sim \mathrm{Dir}(\gamma_2 \boldsymbol{p}')$ and then $\boldsymbol{q}'$ is serving as the probability vector that we use to randomly select clients in each round. As we note before, a smaller value of *concentration* parameter γ_2 leads to a higher variation of $\boldsymbol{q}'$ and thereby causing greater unbalance in selection. Fig. 5 show how the model accuracy evolve with different γ_2, under different non-iid extent (given by γ_1). Among which, subfigures (a), (b), and (c) depict that of the training for fashionMnist, where we can see that a higher γ_2 (a fairer selection) boasts a higher final model accuracy. Also, a similar observation, or an even more conspicuous one, can be found in our training for CIFAR-10, as indicated in subfigures (d), (e), and (f). From our result, it appears that the fairness factor might have different degrees of influence for the training of different datasets. More radically, we are in fact guessing that fairness factor would play a more critical role in a more complicated task. Our theory is that training of a harder task might require more diversified data (in terms of both targets and features), and corresponding, the relative information that each piece of data contains would reduce, and thereby, the training of those tasks should better involve as much available data as possible (i.e., better to be fair), so as to improve the model performance (specifically, final accuracy).

On the other hand, although the experimental data does demonstrate a profound impact of non-iid extent on the model stability and convergence speed during training (as we can observe in Fig. 5 that when γ_1 decreases, more jitters on the curve and more rounds underwent before

---

6. Both Cifar and fashion-Mnist have 10 targets (or classes)
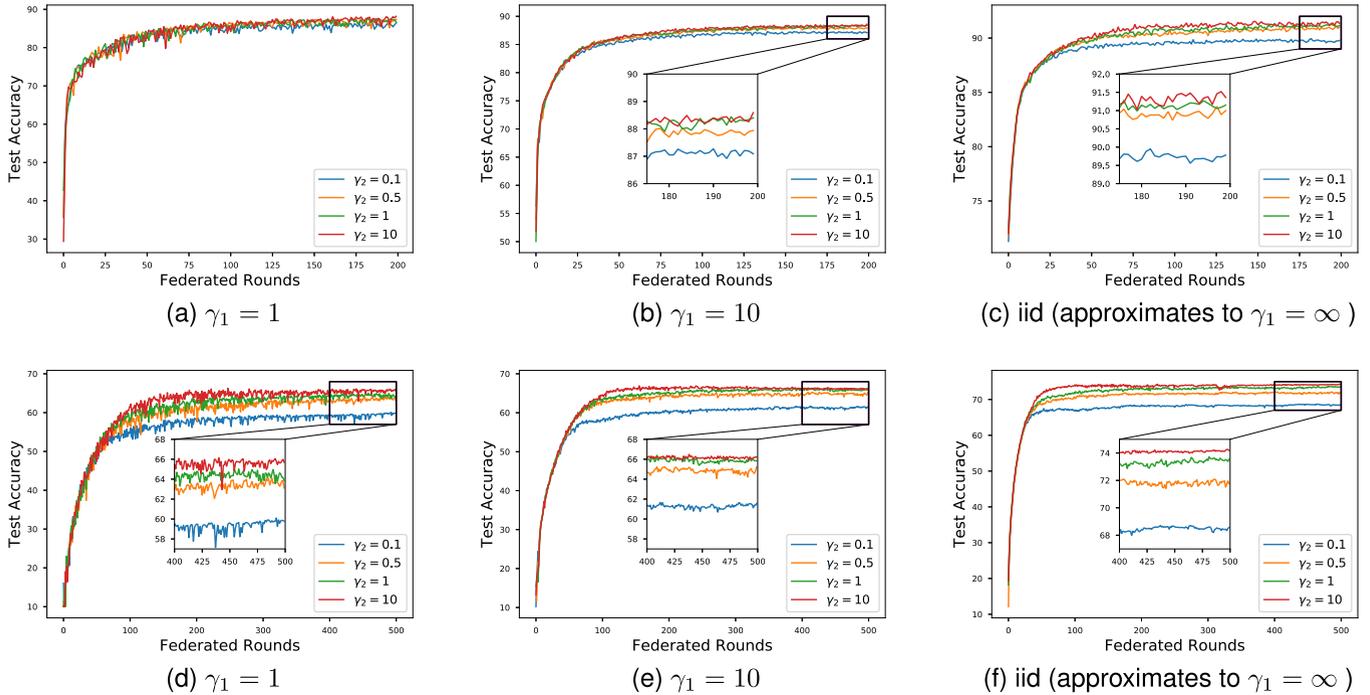
Fig. 5. Fairness impact under fashion-MNIST ((a), (b), and (c)) and CIFAR-10 ((d), (e), and (f)).

convergence), it does not explicitly show the fairness factor (as reflected by γ_2) being more or less influential with the change of γ_1, which seems to tell us that our defined non-iid extent has little or no impact on the effect of fairness.

6.2.3 Accuracy Versus Federated Round

Fig. 6 depicts how our proposed RBCS-F with different settings of V performs in the real training, being compared with the baselines, random and FedCS(3). The result is consistent with our former conjecture that RBCS-F with a smaller V, which is known to be fairer, would achieve a higher final accuracy after rounds of training. Random, a categorically fair scheme, yields the best performance in terms of final model accuracy, while the FedCS(3), another extreme in terms of fairness, does not promise us a commensurate result.
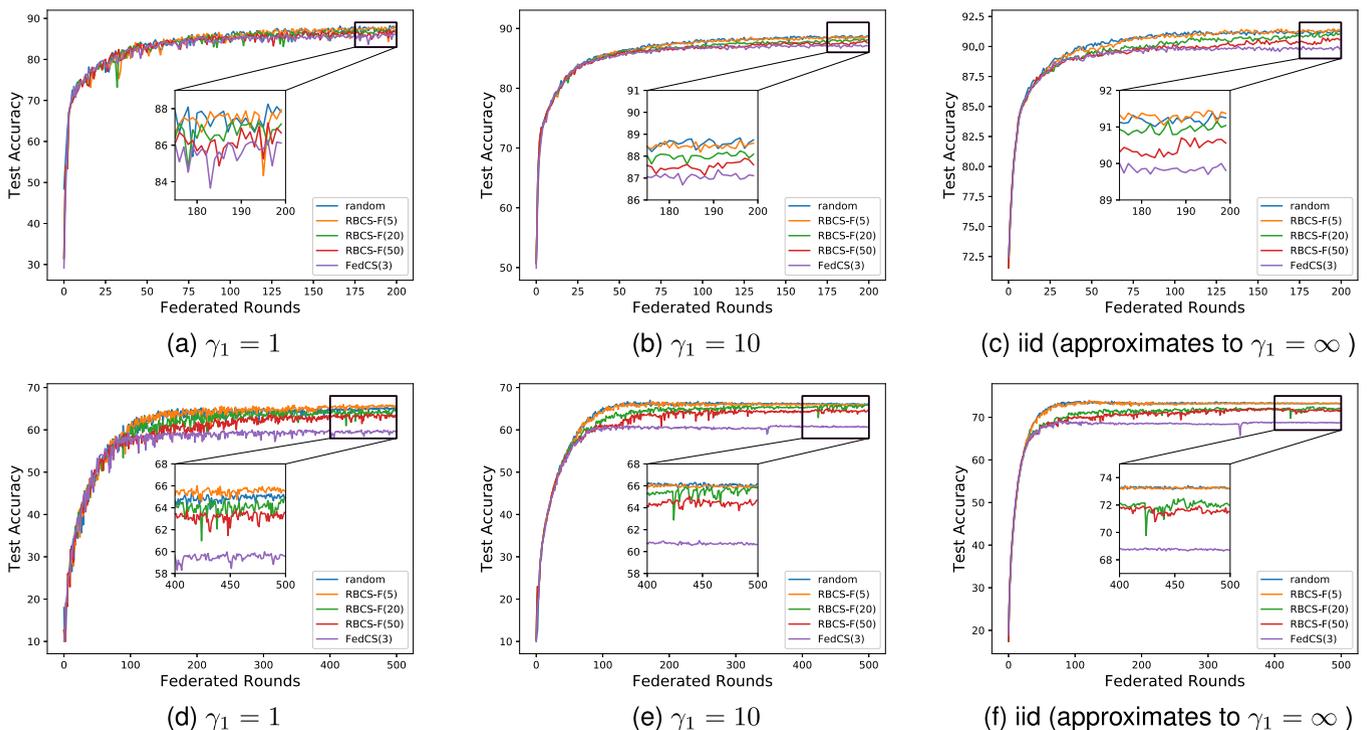
Fig. 6. Accuracy versus federated rounds for fashion-MNIST ((a), (b), (c)) and CIFAR-10 ((d), (e), (f)).

Another point we are interested in is that RBCS-F with a higher penalty seems to spend more rounds to reach a certain accuracy, which we refer to a lower *round efficiency*. This phenomenon can be justified by the result from Fig. 2, which indicates that RBCS-F with a higher penalty tends only to consider fairness when the queue length is large, or in other words, only during a big number of training rounds. Correspondingly, the delay of fairness consideration would make the global model having the chance of aggregating some seldom access data only when the number of rounds is large, and thereby, causing postpone on convergence. Besides, we also found that RBCS-F generally outperforms FedCS(3) in terms of round efficiency, which is conspicuously depicted by subfigures (e) and (f).

6.2.4 Accuracy Versus Training Time

Due to space limit, this section is moved to Appendix E, available in the online supplemental material.

7 CONCLUSION AND FUTURE PROSPECT

In this paper, we have investigated the client selection problem for federated learning. Our concern mainly focuses on the tradeoff between fairness factor and training efficiency. In light of the experiment on our proposed method, we found that fairness is indeed playing a critical role in the training process. In particular, we show that a fairer strategy could promise us a higher final accuracy while inevitably sacrificing a few training efficiency. In terms of how the fairness factor would affect the final achieved accuracy, as well as the convergence speed, however, we could not figure out a rigorous way to quantify their relation. And neither could we track down from the existing literature any theoretical analysis of the fairness factor for FL, making this particular issue quite worthy of investigation. Our future effort would be mainly on this emerging issue.

ACKNOWLEDGMENTS

REFERENCES

[1] 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.

[2] Q. Zeng *et al.*, "Energy-efficient radio resource allocation for federated Edge learning," in *Proc. IEEE Int. Conf. Commun. Workshops*, 2020, pp. 1–6.

[3] F. Li, J. Liu, and B. Ji, "Combinatorial sleeping bandits with fairness constraints," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 3, pp. 1799–1813, Third Quarter 2020.

[4] S. Deng, H. Zhao, J. Yin, S. Dustdar, and A. Y. Zomaya, "Edge intelligence: The confluence of Edge computing and artificial intelligence," 2019, *arXiv: 1909.00560*

[5] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-Edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Netw.*, vol. 33, no. 5, pp. 156–165, Sep./Oct. 2019.

[6] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with Edge computing," *arXiv: 1905.10083*, May 2019. [Online]. Available: http://arxiv.org/abs/1905.10083

[7] Y. Kang *et al.*, "Neurosurgeon: Collaborative intelligence between the cloud and mobile edge," *ACM SIGARCH Comput. Archit. News*, vol. 45, no. 1, pp. 615–629, 2017.

[8] B. McMahan *et al.*, "Communication-efficient learning of deep networks from decentralized data," *Artif. Intell. Statist.*, pp. 1273–1282, 2017.

[9] A. Hard *et al.*, "Federated learning for mobile keyboard prediction," 2018, *arXiv:1811.03604*.

[10] S. Ramaswamy, R. Mathews, K. Rao, and F. Beaufays, "Federated learning for emoji prediction in a mobile keyboard," 2019, *arXiv: 1906.04329*.

[11] Y. Liu *et al.*, "Fedvision: An online visual object detection platform powered by federated learning," 2020, *arXiv: 2001.06202*.

[12] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data," 2018, *arXiv: 1811.11479*.

[13] L. Liu *et al.*, "Client-edge-cloud hierarchical federated learning," in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–6.

[14] N. Yoshida *et al.*, "Hybrid-FL for wireless networks: Cooperative learning mechanism using non-IID data," in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–7.

[15] D. Ye, R. Yu, M. Pan, and Z. Han, "Federated learning in vehicular edge computing: A selective model aggregation approach," *IEEE Access*, vol. 8, pp. 23 920–23 935, 2020.

[16] H. H. Yang, Z. Liu, T. Q. Quek, and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 317–333, Jan. 2020.

[17] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," 2019, *arXiv: 1903.03934*.

[18] W. Wu, L. He, W. Lin, S. Jarvis, R. Mao, and C. Maple, "SAFA: A semi-asynchronous protocol for fast federated learning with low overhead," 2019, *arXiv: 1910.01355*.

[19] J. Kang, Z. Xiong, D. Niyato, H. Yu, Y.-C. Liang, and D. I. Kim, "Incentive design for efficient federated learning in mobile networks: A contract theory approach," in *Proc. IEEE VTS Asia Pacific Wireless Commun. Symp.*, 2019, pp. 1–5.

[20] L. U. Khan *et al.*, "Federated learning for edge networks: Resource optimization and incentive mechanism," *IEEE Commun. Mag.*, vol. 58, no. 10, pp. 88–93, 2020.

[21] E. Bagdasaryan *et al.*, "How to backdoor federated learning," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2938–2948.

[22] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 661–670.

[23] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári, "Improved algorithms for linear stochastic bandits," in *Proc. Advances Neural Inf. Process. Syst.*, 2011, pp. 2312–2320.

[24] L. Qin, S. Chen, and X. Zhu, "Contextual combinatorial bandit and its application on diversified online recommendation," in *Proc. SIAM Int. Conf. Data Mining*, 2014, pp. 461–469.

[25] S. Li, B. Wang, S. Zhang, and W. Chen, "Contextual combinatorial cascading bandits," in *Proc. 33rd Int. Conf. Int. Conf. Mach. Learn.*, 2016, pp. 1245–1253.

[26] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures Commun. Netw.*, vol. 3, no. 1, pp. 1–211, 2010.

[27] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," 2019, *arXiv: 1909.06335*

Tiansheng Huang is currently working toward the MS degree with the School of Computer Science and Engineering, South China University of Technology, China. His research interests include system design, distributed computing and optimization theory.

Weiwei Lin (Member, IEEE) received the BS and MS degrees from Nanchang University, in 2001 and 2004, respectively, and the PhD degree in computer application from the South China University of Technology, in 2007. He has been serving as visiting scholar with Clemson University, from 2016 to 2017. Currently, he is a professor with the School of Computer Science and Engineering, South China University of Technology. His research interests include distributed systems, cloud computing, big data computing and AI application technologies. He has published more than 100 papers in refereed journals and conference proceedings. He has been the reviewers for many international journals, including *IEEE Transactions on Computers*, *IEEE Transactions on Services Computing*, *IEEE Transactions on Cloud Computing*, *IEEE Transactions on Sustainable Computing*, *IEEE Information Sciences*, *IEEE Future Generation Computer Systems*, etc. He is a senior member of CCF.

Wentai Wu received the bachelor's and master's degrees in computer science from the South China University of Technology, in 2015 and 2018, respectively. Currently, he is working toward the PhD degree supervised by Dr. Ligang. He with the Department of Computer Science, the University of Warwick, United Kingdom. His research interests include parallel and distributed computing, distributed machine learning, and energy-efficient computing.
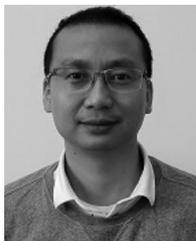
Ligang He received the PhD degree in computer science from the University of Warwick, United Kingdom, and worked as a post-doctoral researcher with the University of Cambridge, United Kingdom. He is currently a reader with the Department of Computer Science, the University of Warwick. He has published more than 130 articles in journals and conferences, such as the *IEEE Transactions on Computers*, *IEEE Transactions on Parallel and Distributed Systems*, TACO, IPDPS, ICPP, SC, VLDB. His research interests include parallel and distributed processing, cluster, high performance computing, and cloud computing.

Keqin Li (Fellow, IEEE) is a SUNY distinguished professor of computer science with the State University of New York. He is also a distinguished professor with Hunan University, China. His current research interests include cloud computing, fog computing and mobile edge computing, energy-efficient computing and communication, embedded systems and cyber-physical systems, heterogeneous computing systems, big data computing, high performance computing, CPU-GPU hybrid and cooperative computing, computer architectures and systems, computer networking, machine learning, intelligent and soft computing. He has published more than 640 journal articles, book chapters, and refereed conference papers, and has received several best paper awards. He is currently or has served on the editorial boards of the *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Computers*, *IEEE Transactions on Cloud Computing*, *IEEE Transactions on Services Computing*, and *IEEE Transactions on Sustainable Computing*.

Albert Y. Zomaya (Fellow, IEEE) is currently the chair professor of High Performance Computing & Networking with the School of Information Technologies, The University of Sydney. He is also the director in the Centre for Distributed and High Performance Computing, which was established in late 2009. He published more than 500 scientific papers and articles and is a author, co-author or editor of more than 20 books. He served as the editor-in-chief of the *IEEE Transactions on Computers* (2011–2014). He serves as an associate editor for 22 leading journals, such as, the *ACM Computing Surveys, IEEE Transactions on Computational Social Systems*, *IEEE Transactions on Cloud Computing*, and *Journal of Parallel and Distributed Computing*. He delivered more than 150 keynote addresses, invited seminars, and media briefings and has been actively involved, in a variety of capacities, in the organization of more than 600 national and international conferences. He received the IEEE Technical Committee on Parallel Processing Outstanding Service Award (2011), the IEEE Technical Committee on Scalable Computing Medal for Excellence in Scalable Computing (2011), and the IEEE Computer Society Technical Achievement Award (2014). He is a chartered engineer, a fellow of AAAS, IEEE, and IET (United Kingdom). His research interests include areas of parallel and distributed computing and complex systems.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.