Toward Sustainable AI: Federated Learning Demand Response in Cloud-Edge Systems via Auctions

Fei Wang[1], Lei Jiao[2], Konglin Zhu[1,3], Xiaojun Lin[4], Lei Li[1]
[1]Beijing University of Posts and Telecommunications, China [2]University of Oregon, USA
[3]Purple Mountain Laboratories, China [4]Purdue University, USA

***Abstract*—Cloud-edge systems are important Emergency Demand Response (EDR) participants that help maintain power grid stability and demand-supply balance. However, as users are increasingly executing artificial intelligence (AI) workloads in cloud-edge systems, existing EDR management has not been designed for AI workloads and thus faces the critical challenges of the complex trade-offs between energy consumption and AI model accuracy, the degradation of model accuracy due to AI model quantization, the restriction of AI training deadlines, and the uncertainty of AI task arrivals. In this paper, targeting Federated Learning (FL), we design an auction-based approach to overcome all these challenges. We firstly formulate a non-linear mixed-integer program for the long-term social welfare optimization. We then propose a novel algorithmic approach that generates candidate training schedules, reformulates the original problem into a new schedule selection problem, and solves this new problem using an online primal-dual-based algorithm, with a carefully embedded payment design. We further rigorously prove that our approach achieves truthfulness and individual rationality, and leads to a constant competitive ratio for the long-term social welfare. Via extensive evaluations with real-world data and settings, we have validated the superior practical performance of our approach over multiple alternative methods.**

I. Introduction

Distributed cloud-edge systems often consume an enormous amount of energy from the power grid and are well situated for Emergency Demand Response (EDR) programs [1]–[3]. Typically, during an EDR period, the power grid sends signals with a time-varying energy cap to the cloud-edge system, and the latter must reduce its energy consumption to under the cap, which helps ensure the stability and demand-supply balance of the power grid. To do so, the cloud-edge system operator needs to carefully manage its workload during EDR periods, especially in the case of a public or shared environment where different users submit workloads to the system. Yet, each user often only cares about the execution of her own workload, without accounting for the energy consumption or EDR of the entire system—an issue known as "split incentives".

One approach to address split incentives is based on auctions [3]–[5]. That is, the cloud-edge operator acts as the auctioneer, and each user acts as a bidder that submits a bid with her own valuation (i.e., amount of money to pay) for executing her workload or task. The auctioneer then strategically selects the winning bids considering EDR, and schedules the execution of the corresponding tasks. Compared with asking the cloud-edge operator to set prices on users' tasks dynamically, which could be tricky due to possible over-/under-pricing, auctions can make users "EDR-aware" (e.g., when deciding bid valuations), avoid mis-pricing for the cloud-edge operator, increase the cloud-edge operator's profit, and achieve market efficiency and agility through real-time demand and supply [6].

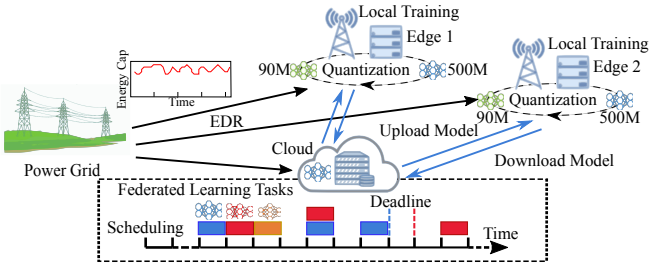

Fig. 1: System scenario

Unfortunately, the emerging paradigm shift that users are increasingly executing artificial intelligence (AI) or machine learning (ML) workloads makes the auction approach much more difficult, due to multiple fundamental challenges.

First, the unique features of ML workloads significantly complicate the cloud-edge management under EDR. For example, training an ML model in a distributed manner, such as federated learning (FL) [7], [8], in the cloud-edge system has unique computation and communication patterns due to local training on edge servers and global aggregations in the cloud, involving complex trade-offs among energy consumption and model accuracy. ML models can also be quantized [9], i.e., updated, transferred, and stored in lower bitwidth, which can save energy but impact the model accuracy. Users may enforce deadlines for completing their ML tasks [10], intertwined with energy and accuracy as well. All these complexities, as shown in Fig. 1, need to be considered in the auction design.

Second, to pursue the long-term optimization, whether to accept the bid and how to spread the training over future time slots need to be determined dynamically in an online manner in response to each ML task arrival. For example, if some bids are accepted and scheduled for execution, then the system may not be able to accept other bids of higher values in the future

This work was supported in part by the National Key Research and Development Program of China (No. 2022YFB2503202), in part by the Beijing Natural Science Foundation (4222033), in part by the National Natural Science Foundation of China (No. 62176024), in part by Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing, and in part by the U.S. National Science Foundation (CNS-2047719, CNS-2225949, CNS-2113893, ECCS-2129631, and CNS-2225950). Corresponding authors are Lei Jiao and Lei Li.

due to the energy cap and resource constraints; further, ML model training needs to choose between more training per time slot over fewer time slots and less training per time slot over more time slots. Any decision now will set an irrevocable baseline for future decisions; as what bids or tasks will arrive and what decisions will be made in the future are unknown, making good decisions now on the fly is challenging.

Third, the auction mechanism also needs to attain the desired economic properties of truthfulness (i.e., a bid maximizes its utility when bidding its true valuation) and individual rationality (i.e., every bid achieves non-negative utility) via careful payment design. Here, the utility of a bid refers to the difference between its true valuation and its payment, incorporating the deadline-violation penalty. Classic approaches such as Vickrey-Clarke-Groves (VCG) auctions [11], [12] require computing auction outcomes exactly, which are infeasible in our case due to the intractability of our problem, as described below. Also, unlike typical VCG situations, the bids in our case arrive sequentially, instead of all at once, and are equipped with adaptable quantization and execution scheduling, instead of fixed commitments. These hinder the application of VCG.

Existing research is limited and insufficient for the scenario targeted in this paper. None of the cloud and/or edge EDR works [1]–[3], [13], [14] have considered AI/ML workloads. Meanwhile, FL optimization [8], [15]–[18] is often focused on a single FL system or an offline setting, and does not consider *multiple* FL tasks in the *online* setting, not to mention auctions. ML/FL task scheduling [19]–[23] also falls short due to lack of EDR and fine-grained energy control. To the best of our knowledge, this paper is the first to study AI/ML EDR in terms of the online scheduling of FL tasks with quantization and deadlines via auctions. See Section VI for detailed discussions.

In this paper, targeting FL, we firstly model and formulate the continuous auction over time as a long-term social welfare maximization problem, capturing the utility of both the cloud-edge system and that of the FL tasks. Our formulation incorporates all the factors of FL computation/communication patterns, model quantization, and training deadlines, while guaranteeing the specified model accuracy and respecting EDR energy caps and resource constraints. Our models make no assumption on FL task and EDR dynamics, and are compatible with both hard and soft deadlines. Our problem turns out to be a non-linear mixed-integer program, which is NP-hard.

To solve this problem, we then design novel polynomial-time online approximation algorithms. Our approach features a non-trivial problem reformulation, converting the complicated schedule computing problem for the FL tasks to an equivalent but relatively "easier" schedule selection problem via expanding and filtering out the candidate schedules and "absorbing" the non-linearity. For this new problem which is still intractable, we explore online primal-dual optimization [24] to design an online algorithm that cautiously maintains feasible solutions to both the primal problem of schedule selection and its dual problem and selects training schedules in a completely online manner without any knowledge about unpredictable future FL tasks. For performance analysis, we rigorously prove that our online algorithm leads to a parameterized-constant competitive ratio. As part of the auction process, our approach embeds payment calculation for each winning bid into the primal-dual design, and meets the conditions as stated in the Myerson theorem [25], based on which we further show truthfulness and individual rationality (assuming deadline-violation penalty functions are always truly reported).

Finally, we conduct extensive evaluations for a varying number of FL tasks that dynamically arrive in 168 hours [26] based on real-world training data [27], [28], EDR events [29], electricity prices [30], etc. We observe multiple results: (i) Our approach achieves $\sim 2\times$ social welfare, compared to two greedy bid selection and scheduling approaches and a state-of-the-art scheduling approach, and is more advantageous when deadline violation becomes more penalized; (ii) Quantization needs careful control—increasing the bit precision can save energy by reducing training iterations for attaining the desired accuracy, but higher bit precision, i.e., exceeding 15 bits in our case, can also increase energy due to the growth of the model size in bits; (iii) Our approach achieves a relatively stable empirical competitive ratio, reaching at least about $2/3$ of the offline maximal social welfare; (iv) Our approach attains truthfulness and individual rationality in practice; (v) Our approach is efficient and runs fast to produce control decisions, finishing in 25 minutes in total for 500 FL tasks in 168 hours.

II. Modeling and Formulation

A. System Settings and Models

Cloud-Edge System: We consider a system consisting of a cloud and a set $\mathcal{N} = \{1, ..., |\mathcal{N}|\}$ of distributed and potentially heterogeneous edges. Here, an "edge" refers to a micro data center or server cluster co-located with a cellular base station or a WiFi access point. The edges can communicate with the cloud via wireline backhaul networks. We study the entire system over a series of consecutive time slots $\mathcal{T} = \{1, ..., |\mathcal{T}|\}$.

Demand Response: The cloud and the edges are all powered by the electricity grid and join the Emergency Demand Response (EDR) program. That is, as the EDR signal comes, the cloud-edge system will receive an energy cap for each time slot E_t, $\forall t \in \mathcal{T}$, and must reduce the energy consumed (by the federated learning tasks, as described next) from the power grid to under the gap at each corresponding time slot. There could be multiple EDR periods intermittently spreading over the time horizon $\mathcal{T}$; we can think of $E_t = +\infty$ for any t when no EDR occurs. We can actually introduce the following function $f_t(\cdot)$ to capture the energy cost at the time slot t:

$$f_t(e_t) = \begin{cases} h_t e_t & \text{if} \quad e_t \le E_t \\ +\infty & \text{otherwise} \end{cases},$$

where h_t is the electricity price of the grid and e_t is the amount of electricity consumed by the cloud-edge system at t.

Federated Learning (FL) Tasks: We consider a set $\mathcal{I} = \{1, ..., |\mathcal{I}|\}$ of federated learning (FL) tasks. An FL task $i \in \mathcal{I}$ is defined as $A_i = \left\{t_i, \varepsilon_i, \{\mathcal{D}_{i,k}, \forall k \in \mathcal{N}\}, L_i\right\}$, where t_i is the time slot when the FL task i arrives at the system; ε_i is convergence accuracy of the global model that is to

be trained of the FL task i; $\mathcal{D}_{i,k}$ refers to the set of the training data located at the edge k for the FL task i, with each data sample $n \in \mathcal{D}_{i,k}$ corresponding to the loss function $f_n(\cdot)$; and L_i is the required number of "local iterations" that need to be executed in each "global iteration" during training, which will be elaborated next. If we use $\boldsymbol{w}_i$ to represent the global model to be trained in the FL task i, then the local loss at the edge k is $F_{i,k}(\boldsymbol{w}_i) = \frac{1}{|\mathcal{D}_{i,k}|}\sum_{n\in\mathcal{D}_{i,k}} f_n(\boldsymbol{w}_i)$, and the global loss is $F_i(\boldsymbol{w}_i) = \frac{1}{\sum_k |\mathcal{D}_{i,k}|}\sum_k\big(|\mathcal{D}_{i,k}| \cdot F_{i,k}(\boldsymbol{w}_i)\big)$. The optimal model for the FL task i can be thus defined as $\boldsymbol{w}_i^* = \arg\min_{\boldsymbol{w}_i} F_i(\boldsymbol{w}_i)$. While $\boldsymbol{w}_i^*$ remains unknown, our goal here is to find out $\boldsymbol{w}_i$ that satisfies $F_i(\boldsymbol{w}_i) - F_i(\boldsymbol{w}_i^*) \leq \varepsilon_i$ in a distributed manner which is further described below.

FL with Quantization: To control the energy consumption of FL, we consider "quantization" in this paper. That is, for the FL task i, we properly use $n_i \in \{1, ..., n_{\max}\}$ bits to store every single parameter of the model to be trained, where $n_{\max}$ is pre-specified. Without loss of generality, we consider fixed-point quantization here [31]. We use a stochastic quantization scheme, i.e., for any given model parameter w, we quantize it by changing w to $Q(w)$ so that we use n_i bits to store $Q(w)$:

$$Q(w) = \begin{cases} \lfloor w \rfloor, & \text{with probability } \frac{\lfloor w \rfloor + \jmath - w}{\jmath} \\ \lfloor w \rfloor + \jmath, & \text{with probability } \frac{w - \lfloor w \rfloor}{\jmath} \end{cases},$$

where $\jmath = 2^{-n_i+1}$; and here $\lfloor w \rfloor$ refers to the largest integral multiple of $\jmath$ that is no greater than w.

We now present an example algorithm of FL with quantization for an FL task i. For this FL task, suppose we have selected a set of (unnecessarily continuous) time slots $\mathcal{T}_i \subseteq \mathcal{T}$, which we further represent as $\mathcal{T}_i = \{\mathcal{T}_i[1], \mathcal{T}_i[2], ..., \mathcal{T}_i[|\mathcal{T}_i|]\}$, to conduct the model training process. Then, at each time slot $t \in \mathcal{T}_i$, we perform $r_i \in \{1, ..., r_{\max}\}$ "global iterations", as shown by Algorithm 0. That is, in each global iteration κ, each edge $k \in \mathcal{N}_{i,t}$ downloads the global model from the cloud, updates the downloaded model via the L_i "location iterations", and eventually uploads the updated model to the cloud and lets the cloud conduct the global aggregation. In each local iteration χ, without loss of generality, we could use the mini-batch Stochastic Gradient Descent (SGD) approach to update the model. That is, we do $\boldsymbol{w}^k_{i,t,\kappa,\chi} = \boldsymbol{w}^k_{i,t,\kappa,\chi-1} - \frac{\eta_t}{|\xi^k_{i,t,\chi}|}\sum_{n\in\xi^k_{i,t,\chi}} \nabla f_n\big(Q(\boldsymbol{w}^k_{i,t,\kappa,\chi-1})\big)$, where $\boldsymbol{w}^k_{i,t,\kappa,\chi}$ is the local model produced in the local iteration χ of the global iteration κ at the edge k at the time slot t for the task i; η_t is the learning rate; and $\xi^k_{i,t,\chi}$ represents the one mini-batch of data samples selected from $\mathcal{D}_{i,k}$ for the local iteration, where one just uses standard approaches to divide $\mathcal{D}_{i,k}$ into multiple mini-batches. $\boldsymbol{w}^k_{i,t,\kappa}$ is the local model produced in the global iteration κ at the edge k, and $\boldsymbol{w}_{i,t,\kappa}$ is the global model produced by the global iteration κ. Note the lines in Algorithm 0 where we apply the quantization function $Q(\cdot)$.

We highlight that, in order to achieve $F_i(\boldsymbol{w}_i) - F_i(\boldsymbol{w}_i^*) \leq \varepsilon_i$, the number of global iterations, i.e., $r_i \cdot |\mathcal{T}_i|$, needs to be $r_i|\mathcal{T}_i| \geq \frac{\Upsilon\vartheta_i}{2\varepsilon_i} - \gamma$, given $|\mathcal{N}|$, K_i, L_i and n_i, with only common and mild assumptions for $F_i(\cdot)$. For readers' reference, Υ is a constant related to smoothness; γ is also a constant; and $\vartheta_i = \sum_{k=1}^{|\mathcal{N}|}\frac{\sigma_k^2}{|\mathcal{N}|^2} + \frac{\varpi_i}{2^{2n_i}}(1+\frac{2L_i\varrho^2}{K_i}) + 4(L_i-1)^2\varrho^2 + \frac{4(|\mathcal{N}|-K_i)}{K_i(|\mathcal{N}|-1)}L_i^2\varrho^2$, where σ_k and ϱ relate to gradients, and ϖ_i is a constant [32].

Example Algorithm of FL for Task i

▷ **Input:** K_i, t_i, L_i, n_i, r_i, initial model $\widetilde{\boldsymbol{w}}_i$, $\mathcal{T}_i$;
select randomly the edges $\mathcal{N}_{i,t}$, where $|\mathcal{N}_{i,t}| = K_i$, $\forall t \geq t_i$;
$\boldsymbol{w}_{i,\mathcal{T}_i[1],0} = \widetilde{\boldsymbol{w}}_i$;
invoke **Algorithm 0** for task i at time slot $\mathcal{T}_i[1]$;
for $t \in \{2, 3, ..., |\mathcal{T}_i|\}$ **do**
  $\boldsymbol{w}_{i,\mathcal{T}_i[t],0} = \boldsymbol{w}_{i,\mathcal{T}_i[t-1],r_i}$;
  invoke **Algorithm 0** for task i at time slot $\mathcal{T}_i[t]$;

Algorithm 0: Quantized FL for Task i at Time Slot t

▷ **Input:** L_i, n_i, r_i, $\mathcal{N}_{i,t}$;
for $\kappa = 1, 2, ..., r_i$ **do**
  ▷ **Local training on each edge $k \in \mathcal{N}_{i,t}$:**
  download $\boldsymbol{w}_{i,t,\kappa-1}$ from the cloud;
  $\boldsymbol{w}^k_{i,t,\kappa,0} = \boldsymbol{w}_{i,t,\kappa-1}$;
  for $\chi = 1, 2, ..., L_i$ **do**
    $\boldsymbol{w}^k_{i,t,\kappa,\chi} = \boldsymbol{w}^k_{i,t,\kappa,\chi-1} - \frac{\eta_t}{|\xi^k_{i,t,\chi}|}\sum_{n\in\xi^k_{i,t,\chi}} \nabla f_n\big(Q(\boldsymbol{w}^k_{i,t,\kappa,\chi-1})\big)$;
  $\boldsymbol{w}^k_{i,t,\kappa} = Q(\boldsymbol{w}^k_{i,t,\kappa,L_i})$;
  upload $\boldsymbol{w}^k_{i,t,\kappa}$ to the cloud;
  ▷ **Global aggregation on the cloud:**
  $\boldsymbol{w}_{i,t,\kappa} = Q(\frac{1}{K_i}\sum_{k\in N_{i,t}} \boldsymbol{w}^k_{i,t,\kappa})$;

FL Energy: First, for the global aggregations in the cloud, we denote the energy per aggregation as $E_i''(n_i) = \varphi M_i n_i$, where φ is the energy for computing the per-bit aggregation; M_i is the size of the model in terms of the total number of model parameters; n_i is the number of bits in quantization used to represent each model parameter. Second, for communicating the model between the cloud and the edges, we analogously write the energy per global iteration as $E'_{i,k}(n_i) = \varphi'_k M_i n_i$ [33], where, in contrast to $E_i''(n_i)$, φ'_k is the energy for transferring one bit of the model from or to the edge k. Third, for the local training at each edge k, we use $E_{i,k}(n_i)$ to refer to the energy consumption per local iteration. We have $E_{i,k}(n_i) = \mathcal{O}({n_i}^{\beta_k})$, where $1 < \beta_k < 2$, for computing the SGD update upon a typical two-dimensional processing chip at the edge k [34]. For the ease of the presentation in this paper, we have omitted the exact form of $E_{i,k}(n_i)$ [35].

Auctions: In our auction mechanism, the cloud-edge system (or its operator) acts as the auctioneer and the FL tasks (or their owners) act as the bidders. Upon arrival, the FL task i submits a bid $B_i = \{t_i, \varepsilon_i, L_i, d_i, b_i, g_i(\cdot)\}$, where t_i, ε_i, and L_i are as in A_i and have already been explained previously; d_i is the deadline required for completing the model training for this task; and b_i refers to the bidding price, i.e., the amount of money the FL task i would like to pay if the training is finished before or by d_i; and $g_i(\cdot)$ refers to the penalty function for the deadline violation. For more flexibility and better social welfare, we allow deadline violation in this work. We define

$$g_i(\tau_i) = \begin{cases} g_{c_i}(\tau_i) & \text{if} \quad \tau_i \in \{0, 1, ..., |\mathcal{T}| - d_i\} \\ 0 & \text{otherwise} \end{cases},$$

where τ_i represents the number of time slots for which the completion of training exceeds the deadline d_i, and $g_{c_i}(\cdot)$ is a

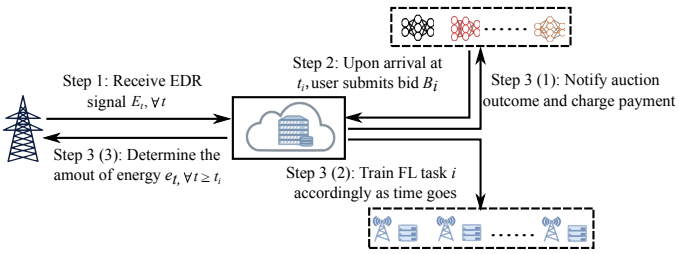

Fig. 2: Illustration of our auction model

non-decreasing function provided by the FL task i. Now, note that $d_i + \tau_i$ is the FL task i's time of training completion; and $b_i - g_i(\tau_i)$ is the bidding price for training completion before or by $d_i + \tau_i$, i.e., the FL task i would pay less money in the case of deadline violation. In this paper, we assume one FL task issues one and only one bid. Fig. 2 shows how the auction works dynamically upon receiving each bid (except "Step 1", where EDR signals are received before all bids).

Control Decisions: The system operator makes the following control decisions upon receiving the bid i at the time slot t_i: (1) $x_i \in \{1, 0\}$, denoting whether or not to choose the bid i as a winning bid for the auction; (2) $y_i^t \in \{1, 0\}$, $\forall t \geq t_i$, denoting whether or not to train the FL task i at the time slot t; (3) $\tau_i \in \{0, 1, 2, ..., |\mathcal{T}| - d_i\}$, denoting the number of time slots by which the last time slot for training the FL task i has exceeded the desired deadline d_i; (4) $n_i \in \{1, ..., n_{\max}\}$, denoting the number of bits to use for quantizing each model parameter of the FL task i; (5) $r_i \in \{1, ..., r_{\max}\}$, denoting the number of global iterations to conduct for each single time slot for the FL task i; (6) $e_t \geq 0$, $\forall t \geq t_i$, denoting the amount of energy to consume from the power grid at t; and (7) $p_i \geq 0$, denoting the amount of payment made by the bid i. Note that we can dynamically update e_t, $\forall t \geq t_i$ as the bid i arrives.

Utility of Bidders: The utility of each bid or each FL task consists of the corresponding bidding price, considering the potential deadline violation, minus the payment made. So, the total utility of all the bidders is $\sum_{i \in \mathcal{I}}(x_i b_i - g_i(\tau_i) - p_i)$.

Utility of Auctioneer: The utility of the cloud-edge system consists of the received payments minus the energy cost for training all the FL tasks: $\sum_{i \in \mathcal{I}} p_i - \sum_{t \in \mathcal{T}} f_t(e_t)$.

B. Problem Formulation and Algorithm Requirement

Social Welfare Maximization: The "social welfare" refers to the total utility of the entire system. We formulate the social welfare maximization problem $\mathbb{P}$ as follows. Note that the payments are naturally cancelled in the formulation for now, but still need to designed and calculated later.

$$\mathbb{P}: \quad \max \quad P = \sum_{i \in \mathcal{I}} \big(x_i b_i - g_i(\tau_i)\big) - \sum_{t \in \mathcal{T}} f_t(e_t) \quad (1)$$

$$\text{s.t.} \quad y_i^t t \leq d_i + \tau_i, \forall i \in \mathcal{I}, \forall t \geq t_i, \quad (1a)$$

$$\sum_{i \in \mathcal{I}} y_i^t \leq C, \forall t \in \mathcal{T}, \quad (1b)$$

$$y_i^t \leq x_i, \forall i \in \mathcal{I}, \forall t \in \mathcal{T}, \quad (1c)$$

$$\sum_{t \in \mathcal{T}} y_i^t r_i \geq x_i\big(\tfrac{\Upsilon \vartheta_i(n_i)}{2\varepsilon_i} - \gamma\big), \forall i \in \mathcal{I}, \quad (1d)$$

$$\sum_{i \in \mathcal{I}} y_i^t r_i \Big(\sum_{k \in \mathcal{N}_{i,t}} \big(L_i E_{i,k}(n_i) + 2E'_{i,k}(n_i)\big) + E''_i(n_i)\Big) \leq e^t, \forall t \in \mathcal{T}, \quad (1e)$$

Algorithm 1 Online Auction Mechanism for FL Tasks

Input: $\{E_t\}$, C, $\{h_t\}$.

initialize $\widetilde{e}_t = 0$, $m_t = 0$, $v_t = h_t$, $z_t = 0$, $\forall t \in \mathcal{T}$;
for $i = 1, 2, ..., |\mathcal{I}|$ **do**
  collect the bidding information B_i from the bid i;
  select randomly the edges $\mathcal{N}_{i,t}$, where $|\mathcal{N}_{i,t}| = K_i$, $\forall t \geq t_i$;
  invoke **Algorithm 2** to obtain $\widetilde{x}_i$, $\{\widetilde{y}_i^t\}$, $\widetilde{n}_i$, $\widetilde{r}_i$, $\widetilde{p}_i$;
  if $\widetilde{x}_i == 1$ **then**
    accept the bid i, and charge $\widetilde{p}_i$;
    get $\mathcal{T}_i = \{t | \widetilde{y}_i^t = 1, \forall t \geq t_i\}$;
    set initial model $\boldsymbol{w}_{i,\mathcal{T}_i[1],0} = \widetilde{\boldsymbol{w}}_i$;
    invoke **Algorithm 0** for i for L_i, $\widetilde{n}_i$, $\widetilde{r}_i$ and $\mathcal{N}_{i,\mathcal{T}_i[1]}$;
    for $t \in \{2, 3, ..., |\mathcal{T}_i|\}$ **do**
      $\boldsymbol{w}_{i,\mathcal{T}_i[t],0} = \boldsymbol{w}_{i,\mathcal{T}_i[t-1],r_i}$;
      invoke **Algorithm 0** for i for L_i, $\widetilde{n}_i$, $\widetilde{r}_i$ and $\mathcal{N}_{i,\mathcal{T}_i[t]}$;
    end for
  else
    reject the bid i;
  end if
end for

$$x_i \in \{0, 1\}, y_i^t \in \{0, 1\}, n_i \in \{1, ..., n_{\max}\},$$
$$r_i \in \{1, ..., r_{\max}\}, \tau_i \in \{0, 1, ..., |\mathcal{T}| - d_i\},$$
$$e_t \geq 0, \forall k \in \mathcal{N}_{i,t}, \forall i \in \mathcal{I}, \forall t \in \mathcal{T}. \quad (1f)$$

The objective (1) maximizes the total utility of the bidders and the auctioneer. Constraint (1a) ensures that training of an FL task starts after it arrives and ends before or at $d_i + \tau_i$. Constraint (1b) ensures that the number of FL tasks trained simultaneously respects the system capacity C. Constraint (1c) ensures that only the FL tasks that win in the auction can be trained. Constraint (1d) ensures that a sufficient number of global iterations are conducted for each FL task to achieve the target accuracy. Constraint (1e) ensures that sufficient energy is consumed from the power grid for training the FL tasks at each time slot. Constraint (1f) specifies the domains of the control variables. This problem is NP-hard as it contains the knapsack problem as a special case.

Algorithmic Goal: The goal is to design polynomial-time online approximation algorithms to produce control decisions that can lead to a provable competitive ratio

$$r = P^*/P\big(\{\widetilde{x}_i, \{\widetilde{y}_i^t, \forall t\}, \widetilde{\tau}_i, \widetilde{n}_i, \widetilde{r}_i, \forall i\}, \{\widetilde{e}_t, \forall t\}\big).$$

Here, $r \geq 1$. In r, the numerator is P^*, i.e., the value of P evaluated with the offline optimal solutions to $\mathbb{P}$ where all the inputs over the entire time horizon are assumed known at once in advance; and the denominator is the value of P evaluated with $\{\widetilde{x}_i, \{\widetilde{y}_i^t, \forall t\}, \widetilde{\tau}_i, \widetilde{n}_i, \widetilde{r}_i, \forall i\}, \{\widetilde{e}_t, \forall t\}$, which are the solutions produced by our online algorithms. Note that we also need to determine the payment for each bid i to satisfy truthfulness and individual rationality, as described later.

III. Online Algorithm Design

We propose an online auction mechanism, i.e., Algorithm 1, to conduct the auction and execute the training of FL tasks.

The key of Algorithm 1 is the invocation of Algorithm 2 which generates the immediate auction outcome and the controlled schedule of training as each FL task dynamically arrives.

Our technical roadmap to designing Algorithm 2 is to firstly reformulate the original problem $\mathbb{P}$ to an equivalent "schedule selection" problem $\mathbb{P}_1$, then derive the dual problem $\mathbb{D}_1$ for $\mathbb{P}_1$, and finally design an online primal-dual approach to simultaneously solve $\mathbb{D}_1$ and $\mathbb{P}_1$ (and thus the original problem $\mathbb{P}$), while calculating the payment to each bid.

A. Problem Reformulation

In order to solve the original problem $\mathbb{P}$, we reformulate it as a "schedule selection" problem $\mathbb{P}_1$. For each FL task i, we define a "schedule" as a concrete assignment of values to $\{x_i, \{y_i^t, \forall t\}, \tau_i, n_i, r_i\}$. That is, for a given schedule, the decision variables x_i, $\{y_i^t, \forall t\}$, τ_i, n_i, and r_i have concrete values and are now constants. Each schedule for the FL task i is thus a definite decision for i on whether to accept the bid, how to schedule the FL training, how to configure the bit precision for quantization, and how many global iterations to perform per time slot. We use ξ_i to denote the set of all the feasible schedules for the FL task i which satisfy Constraints (1a), (1c), and (1d) in the formulation of $\mathbb{P}$. For the schedule $l \in \xi_i$, if we use $\{y_{i,l}^t, \forall t\}$ to denote the specific constant values taken by the decision variables $\{y_i^t, \forall t\}$, then we can define $\mathcal{T}_{i,l} = \{t | y_{i,l}^t = 1, \forall t \geq t_i\}$ to represent the set of time slots when the FL task i is trained using the schedule l for this FL task. Analogously, for $l \in \xi_i$, we use $\tau_{i,l}$, $n_{i,l}$, and $r_{i,l}$ to represent the specific constant values to configure the decision variables τ_i, n_i, and r_i, respectively; we also use $b_{i,l}$ to denote $b_i - g_i(\tau_{i,l})$. Consequently, based on this concept of "schedule", the problem now becomes selecting one particular schedule for each FL task as all the tasks dynamically arrive.

We present the formulation of the schedule selection problem $\mathbb{P}_1$, based on the formulation of the original problem $\mathbb{P}$.

$$\mathbb{P}_1: \quad \max \quad P_1 = \sum_{i\in\mathcal{I}} \sum_{l\in\xi_i} x_{i,l} b_{i,l} - \sum_{t\in\mathcal{T}} f_t(e_t) \quad (2)$$

$$\text{s.t.} \quad \sum_{l\in\xi_i} x_{i,l} \leq 1, \forall i \in \mathcal{I}, \quad (2a)$$

$$\sum_{i\in\mathcal{I}} \sum_{l\in\xi_i: t\in\mathcal{T}_{i,l}} x_{i,l} \leq C, \forall t \in \mathcal{T}, \quad (2b)$$

$$\sum_{i\in\mathcal{I}} \sum_{l\in\xi_i: t\in\mathcal{T}_{i,l}} x_{i,l} r_{i,l} \Big(\sum_{k\in\mathcal{N}_{i,t}} \big(L_i E_{i,k}(n_{i,l}) + 2E'_{i,k}(n_{i,l})\big) + E''_i(n_{i,l}) \Big) \leq e_t, \forall t \in \mathcal{T}, \quad (2c)$$

$$x_{i,l} \in \{0,1\}, \forall l \in \xi_i, \forall i \in \mathcal{I}; e_t \geq 0, \forall t \in \mathcal{T}. \quad (2d)$$

Note that our decision variables now become $x_{i,l}$ and e_t, where e_t has been described previously; and $x_{i,l} = 1$ if we select the schedule l for the FL task i and $x_{i,l} = 0$ if not. Constraint (2a) ensures that we select at most one schedule for each FL task. Constraints (2b) and (2c) are equivalent to (1b) and (1e), respectively. Constraint (2d) specifies the domains. The problem $\mathbb{P}_1$ is equivalent to the problem $\mathbb{P}$, because a solution to $\mathbb{P}_1$ corresponds to a solution to $\mathbb{P}$, and vice versa.

B. Dual Problem

To solve the problem $\mathbb{P}_1$ in an online manner, we derive the Lagrange dual problem $\mathbb{D}_1$ of the problem $\mathbb{P}_1$, and then design a primal-dual-based online algorithm. To get $\mathbb{D}_1$, we relax $x_{i,l} \in \{0,1\}$ to $x_{i,l} \geq 0$ (and we do not need $x_{i,l} \leq 1$ due to Constraint (2a)), and introduce the non-negative dual variables μ_i, m_t, and v_t for Constraints (2a), (2b), and (2c), respectively. The dual problem $\mathbb{D}_1$ can be written as follows:

$$\mathbb{D}_1: \min \quad D_1 = \sum_{i\in\mathcal{I}} \mu_i + \sum_{t\in\mathcal{T}} \sup_{e_t\geq 0} \{v_t e_t - f_t(e_t)\} + \sum_{t\in\mathcal{T}} m_t C \quad (3)$$

$$\text{s.t.} \quad \mu_i \geq b_{i,l} - \sum_{t\in\mathcal{T}_{i,l}} v_t r_{i,l} \Big(\sum_{k\in\mathcal{N}_{i,t}} \big(L_i E_{i,k}(n_{i,l}) + 2E'_{i,k}(n_{i,l})\big) + E''_i(n_{i,l}) \Big) - \sum_{t\in\mathcal{T}_{i,l}} m_t, \quad (3a)$$

$$\mu_i \geq 0, m_t \geq 0, v_t \geq 0, \forall t \in \mathcal{T}, \forall i \in \mathcal{I}, \quad (3b)$$

where $\sup_{e_t\geq 0} \{v_t e_t - f_t(e_t)\}$ is defined as

$$\sup_{e_t\geq 0} \{v_t e_t - f_t(e_t)\} = \begin{cases} 0, & v_t \leq h_t \\ (v_t - h_t) E_t, & v_t > h_t \end{cases}.$$

C. Online Primal-Dual Algorithm

Our online primal-dual-based algorithm simultaneously obtains and maintains feasible solutions to the primal problem $\mathbb{P}_1$ and the dual problem $\mathbb{D}_1$ and constructs approximate solutions to the original problem $\mathbb{P}$ on the fly. The reasons for adopting the primal-dual design are multi-fold. First, we can determine the values of the discrete decision variables to cope with the NP-hardness by leveraging the Karush-Kuhn-Tucker (KKT) conditions. If we can control the dual variables until the dual inequality constraint (3a) becomes tight, i.e., becoming an equality, then the corresponding primal variable $x_{i,l}$ can be set to a non-zero value, i.e., 1 in our case. This is by following the "complementary slackness" in the KKT conditions. Second, we can update the values of the continuous decision variables e_t on the fly each time when selecting the schedule for a new FL task, and by utilizing the duality that the value of the dual objective (3) is always an upper-bound for the value of primal objective (2), we can ensure that our decisions can lead to a provable competitive ratio, as shown later. Third, we can also control our payment via the dual constraint (3a) to ensure the desired economic properties of truthfulness and individual rationality that will be defined next during this process.

To make (3a) tight, as the dual variable μ_i is non-negative, we can set μ_i as

$$\mu_i = \max\Big(0, \max_{l\in\xi_i} \Big(b_{i,l} - \sum_{t\in\mathcal{T}_{i,l}} m_t - \sum_{t\in\mathcal{T}_{i,l}} v_t r_{i,l} \Big(\sum_{k\in\mathcal{N}_{i,t}} \big(L_i E_{i,k}(n_{i,l}) + 2E'_{i,k}(n_{i,l})\big) + E''_i(n_{i,l}) \Big)\Big)\Big). \quad (4)$$

Then, we also set the dual variables m_t and v_t as

$$m_t = (L - h_t)\Big(\frac{U - h_t}{L - h_t}\Big)^{\frac{z_t}{C}}, \quad (5)$$

$$v_t = h_t, \forall t \in \mathcal{T}, \quad (6)$$

where $U = \max_{i\in\mathcal{I}}\{\frac{b_i r_{\max}}{G_i}\}$, $L = \min_{i\in\mathcal{I}}\{\frac{(b_i - g_i(|\mathcal{T}|-d_i))r_{\max}}{G_i}\}$, and $G_i = \frac{\Upsilon\vartheta_i(n_{\max})}{2\varepsilon_i} - \gamma$. Here, U and L represent the maximum and the minimum costs per global aggregation in the cloud, and z_t represents the total number of FL tasks being trained at the time slot t. We define $z_t^i, \forall t \in \mathcal{T}$ as the total number of FL tasks being trained at t after making the control decisions for the task i, so z_t is the final value of z_t^i after all tasks are processed. We initialize $z_t^0 = 0$, $\forall t \in \mathcal{T}$. If the bid i wins, we update $z_t^i = z_t^{i-1} + 1, \forall t \in \mathcal{T}_{i,l}$, where l is the schedule selected; otherwise, we update $z_t^i = z_t^{i-1}$. m_t and v_t are carefully calculated to serve our theoretical analysis later.

In Algorithm 2, for the FL task i, we find the best schedule by iterating n_i and r_i in Lines 2-35. In Lines 5-9, we find out the set $\mathcal{L}$ of the feasible time slots via capacity and energy constraints. In Line 10, we figure out the number of time slots w_i that we need at least in order to conduct training to achieve the target accuracy ε_i. In Lines 11-16, we construct the first schedule l_0, whose training completion time is t_{w_i}. Line 12 corresponds to (4) as shown previously, and Lines 13-15 renew the value of $\varsigma(t_{w_i})$ if the last training time slot passes the deadline. Then, in Lines 17-29, we iteratively construct the total $|\mathcal{L}| - w_i$ best schedules whose training completion time are the $(w_i{+}1)$-th time slot in $\mathcal{L}$, the $(w_i{+}2)$-th time slot in $\mathcal{L}$, the $(w_i{+}3)$-th time slot in $\mathcal{L}$, and so on, respectively. To ensure that the number of the time slots of a training schedule is w_i, we find out the best schedule which has the earliest training completion time t_c, and replace the specific time slot t where the value of $\varsigma(t)$ is biggest in $\{t_1, t_2, ..., t_{w_i}\}$ by t_c in Lines 24-28. This way, we divide the total $C_{|\mathcal{L}|}^{w_i}$ schedules into $|\mathcal{L}| - w_i$ groups according to the different training completion times, and find out the best schedule for every different training completion time. We figure out the "best of the best" schedules which has the smallest objective value in Line 30. In Lines 31-33, we find the smallest objective value for different $\widetilde{n}_i$ and $\widetilde{r}_i$. In Lines 36-42, we update the values of our primal and dual variables based on the KKT conditions as described previously, and also determine the payment in Line 38.

IV. Performance Analysis

A. Time Complexity and Correctness

Theorem 1 *Our approach terminates in polynomial time and returns a feasible solution to the problem* $\mathbb{P}$.

Proof. **Time Complexity:** Algorithm 2 has $O(n_{\max} r_{\max})$ iterations. In particular, Lines 17-29 have $O(|\mathcal{T}|)$ iterations, where in each iteration Line 24 takes $O(|\mathcal{T}|)$. Thus, overall, Algorithm 2 takes $O(n_{\max} r_{\max} |\mathcal{T}|^2)$.

Correctness: In Algorithm 2, Line 6 makes the solution satisfy (2b) and (2c). For the specific schedule $\hat{l}$ being chosen, Line 37 sets $x_{i,\hat{l}}$ to 1, satisfying (2a). (2d) is naturally satisfied. Thus, we have a feasible solution for $\mathbb{P}_1$. Further, in Algorithm 2, Lines 40 and 41 set m_t and μ_i according to (5) and (4), respectively, and together with (6), make the dual solution satisfy (3a)-(3b), thus feasible for $\mathbb{D}_1$. As stated in Section III, solving $\mathbb{P}_1$ is equivalent to solving $\mathbb{P}$. Thus, we have a feasible solution for the original problem $\mathbb{P}$. □

Algorithm 2 Schedule Generation and Selection for Task i

Input: B_i, $\{E_t\}$, C, $\{\widetilde{e}_t\}$, $\{m_t\}$, $\{v_t\}$, $\{z_t\}$.

1: initialize $\widetilde{P} = +\infty$, $\widetilde{x}_i = 0$, $\{\widetilde{y}_i^t\} = \{0\}$, $\forall t \geq t_i$;
2: **for** $n_i = 1, 2, ..., n_{\max}$ **do**
3: **for** $r_i = 1, 2, ..., r_{\max}$ **do**
4: $\mathcal{L} = \emptyset$, $j = 1$;
5: **for** $t = t_i, t_i + 1, t_i + 2, ..., |\mathcal{T}|$ **do**
6: **if** $z_t^{i-1} + 1 \leq C$ and $r_i\Big(\sum_{k\in\mathcal{N}_{i,t}}\big(L_i E_{i,k}(n_i) + 2E'_{i,k}(n_i)\big) + E''_i(n_i)\Big) + e_t \leq E_t$ **then**
7: $\mathcal{L} = \mathcal{L} \cup \{t\}$;
8: **end if**
9: **end for**
10: $w_i = \lceil(\frac{\Upsilon\vartheta_i(n_i)}{2\varepsilon_i} - \gamma)/r_i\rceil$;
11: let l_0 be the first w_i slots $\{t_1, t_2, ..., t_{w_i}\}$ in $\mathcal{L}$;
12: $\varsigma(t) = v_t r_i\Big(\sum_{k\in\mathcal{N}_{i,t}}\big(L_i E_{i,k}(n_i) + 2E'_{i,k}(n_i)\big) + E''_i(n_i)\Big) + m_t, \forall t \in \{t_1, t_2, ..., t_{w_i}\}$;
13: **if** $t_{w_i} > d_i$ **then**
14: $\varsigma(t_{w_i}) = \varsigma(t_{w_i}) + g_i(t_{w_i} - d_i)$;
15: **end if**
16: $P_0 = \sum_{t\in\mathcal{T}_{i,l_0}}\varsigma(t)$;
17: **while** $w_i + j \leq |\mathcal{L}|$ **do**
18: $l_j = l_{j-1}$;
19: let t_c be the $(w_i + j)$-th time slot in $\mathcal{L}$;
20: $\varsigma(t) = v_t r_i\Big(\sum_{k\in\mathcal{N}_{i,t}}\big(L_i E_{i,k}(n_i) + 2E'_{i,k}(n_i)\big) + E''_i(n_i)\Big) + m_t$, $\forall t \in \{t_1, t_2, ..., t_{w_i}\} \cup \{t_c\}$;
21: **if** $t_c > d_i$ **then**
22: $\varsigma(t_c) = \varsigma(t_c) + g_i(t_c - d_i)$;
23: **end if**
24: $t_m = \arg\max_{t\in\{t_1,...,t_{w_i-1}\}}\varsigma(t)$;
25: **if** $\varsigma(t_{w_i}) < \varsigma(t_m)$ **then**
26: $t_m = t_{w_i}$;
27: **end if**
28: $t_{w_i} = t_c$, $P_j = \sum_{t\in\mathcal{T}_{i,l_j}}\varsigma(t)$, $j = j + 1$;
29: **end while**
30: $\hat{j} = \arg\min_j P_j$, $\hat{P} = P_{\hat{j}}$;
31: **if** $\hat{P} < \widetilde{P}$ **then**
32: $\widetilde{P} = \hat{P}$, $\hat{l} = l_{\hat{j}}$, $\widetilde{n}_i = n_i$, $\widetilde{r}_i = r_i$;
33: **end if**
34: **end for**
35: **end for**
36: **if** $b_i - \widetilde{P} > 0$ **then**
37: $\widetilde{x}_i = 1$, $\widetilde{y}_i^t = 1$, $x_{i\hat{l}} = 1$, $\widetilde{\tau}_i = \max\{\max_{t\in\mathcal{T}_{i,\hat{l}}}\{t - d_i\}, 0\}$, $\forall t \in \mathcal{T}_{i,\hat{l}}$;
38: $\widetilde{p}_i = \sum_{t\in\mathcal{T}_{i,\hat{l}}}\Big(m_t + v_t\widetilde{r}_i\Big(\sum_{k\in\mathcal{N}_{i,t}}\big(L_i E_{i,k}(\widetilde{n}_i) + 2E'_{i,k}(\widetilde{n}_i)\big) + E''_i(\widetilde{n}_i)\Big)\Big)$;
39: $z_t^i = z_t^{i-1} + 1$, $\widetilde{e}_t = \widetilde{e}_t + \widetilde{r}_i\Big(\sum_{k\in\mathcal{N}_{i,t}}\big(L_i E_{i,k}(\widetilde{n}_i) + 2E'_{i,k}(\widetilde{n}_i)\big) + E''_i(\widetilde{n}_i)\Big)\Big)$, $\forall t \in \mathcal{T}_{i,\hat{l}}$;
40: $m_t = (L - h_t)\Big(\frac{U - h_t}{L - h_t}\Big)^{\frac{z_t^i}{C}}$, $\forall t \in \mathcal{T}_{i,\hat{l}}$;
41: $\mu_i = b_i - \widetilde{P}$;
42: **end if**
43: **return** $\widetilde{x}_i$, $\{\widetilde{y}_i^t\}$, $\widetilde{\tau}_i$, $\widetilde{n}_i$, $\widetilde{r}_i$, $\{\widetilde{e}_t\}$, $\widetilde{p}_i$, $\{m_t\}$, $\{z_t\}$.

B. Truthfulness and Individual Rationality

We formally define the utility of a bid, based on which we further define truthfulness and individual rationality. We then prove that our approach achieves both of these economic properties. Truthfulness ensures a bid uses its true valuation and has no motivation to lie about its bidding price, and individual rationality ensures there is no loss for each bid no matter it wins or loses in the auction. We assume the deadline-violation penalty functions are always truly reported.

Definition 1 *Utility: The utility of the bid i is*

$$u_i = \begin{cases} v_i - g_i(\tau_i) - p_i, & \text{if } x_i = 1 \\ 0, & \text{otherwise'} \end{cases} \tag{7}$$

where v_i is the true valuation of the bid i if the FL training is completed before the deadline d_i; $g_i(\tau_i)$ is the penalty for training that exceeds d_i; and p_i is the payment.

Definition 2 *Truthfulness: An auction achieves truthfulness if bidding the true valuation maximizes the utility for any bid, i.e., for any b_i, where $b_i \neq v_i$, we have $u_i(v_i - g_i(\tau_i)) \geq u_i(b_i - g_i(\tau_i))$.*

Definition 3 *Individually Rationality: A bid always has non-negative utility regardless of the auction outcome, i.e., for any bid i, we always have $u_i \geq 0$.*

Theorem 2 *According to the Myerson theorem [25], an auction is truthful if and only if (i) the auction outcome is monotone, i.e., for any bid i with bidding prices $b_{i,l}$ and $b'_{i,l}$, if $b_{i,l} \geq b'_{i,l}$, $\forall l$, then bidding $b'_{i,l}$ wins the auction implies that bidding $b_{i,l}$ will also win; and (ii) the winning bid pays the "critical payment", i.e., if the bid i wins with the bidding price $\widetilde{b}_{i,l}$ and the payment $\widetilde{p}_i$, then it will also win if it bids $b_{i,l} > \widetilde{p}_i$, while all the other inputs remain the same. Our approach satisfies both conditions and thus achieves truthfulness.*

Proof. **Monotonicity**: Based on (4) and Algorithm 2, we have

$$\mu_i = \max\Big(0, \max_{l\in\xi_i}\Big(b_{i,l} - \textstyle\sum_{t\in\mathcal{T}_{i,l}} m_t - \sum_{t\in\mathcal{T}_{i,l}} v_t \\ r_{i,l}\Big(\sum_{k\in\mathcal{N}_{i,t}}\big(L_i E_{i,k}(n_{i,l}) + 2E'_{i,k}(n_{i,l})\big) + E''_i(n_{i,l})\Big)\Big)\Big), \forall i.$$

That is, if $b_{i,l} \geq b'_{i,l}$, $\forall l$ and all the other inputs do not change, then correspondingly we have $\mu_i \geq \mu'_i$. If $\mu'_i > 0$, then we have $x_i = 1$; as $\mu_i \geq \mu'_i$, we have $\mu_i > 0$, and still $x_i = 1$.

Critical Payment: Suppose the bid i wins the auction when bidding $\widetilde{b}_{i,l}$ and paying $\widetilde{p}_i$. From Algorithm 2, we have the following: if $b_i - \widetilde{P} = \widetilde{b}_{i,l} - \widetilde{p}_i > 0$, then $x_i = 1$; if $b_i - \widetilde{P} \leq 0$, then $x_i = 0$. Therefore, if $b_{i,l} > \widetilde{p}_i$, we have $x_i = 1$. □

Theorem 3 *Our approach achieves individual rationality.*

Proof. According to the definition of utility, if a bid i loses in the auction, then $u_i = 0$. If a bid i wins in the auction, then we have $x_i = 1$ and $u_i = v_i - g_i(\widetilde{\tau}_i) - \widetilde{p}_i \geq b_i - g_i(\widetilde{\tau}_i) - \widetilde{p}_i = b_i - g_i(\widetilde{\tau}_i) - \sum_{t\in\mathcal{T}_{i,\hat{l}}} m_t - \sum_{t\in\mathcal{T}_{i,\hat{l}}} v_t \widetilde{r}_i\Big(\sum_{k\in\mathcal{N}_{i,t}}\big(L_i E_{i,k}(\widetilde{n}_i) + 2E'_{i,k}(\widetilde{n}_i)\big) + E''_i(\widetilde{n}_i)\Big) = \mu_i > 0$. In this chain, we reach the first inequality due to truthfulness; we further reach the second equality due to how we calculate the payment in Algorithm 2; finally, because $x_i = 1$, we can directly reach the third equality based on (4). Thus, the utility is non-negative. □

C. Competitive Ratio

Theorem 4 *Our approach leads to*

$$P^*/P\big(\{\widetilde{x}_i, \{\widetilde{y}_i^t, \forall t\}, \widetilde{\tau}_i, \widetilde{n}_i, \widetilde{r}_i, \forall i\}, \{\widetilde{e}_t, \forall t\}\big) \leq \alpha,$$

where $\alpha = \max_{t\in\mathcal{T}} \ln(\frac{U-h_t}{L-h_t})$, and U, L, and $\{h_t, \forall t\}$ are as defined previously.

Proof. See details in the Appendix. A key part of the proof is organizing P_1 and D_1 into an inequality based on both the weak duality [24], [36] and the online updates of the primal and the dual variables after scheduling each FL task. □

V. Experimental Study

A. Experimental Settings

FL Tasks: We consider $|\mathcal{T}| = 168$ consecutive time slots, where each time slot equals one hour [37]. We consider image classification FL tasks. We use the MNIST dataset [27] and the CIFAR-10 dataset [28]. We consider two models: (i) the LeNet-5 [38], and (ii) a Convolutional Neural Network (CNN) with two 3×3 convolutional layers (where the first layer has 16 channels and the second layer has 32 channels), with each of them followed by ReLU activation and 2×2 max pooling, a fully-connected layer, and a softmax output layer. We have two datasets and two models, a total of four types of FL tasks.

We consider $100 \sim 500$ FL tasks in total, with each type occupying a quarter of all FL tasks. The number of FL tasks that arrive in each time slot is set in proportion to the dynamic job arrival trace of Google clusters [26], where the last FL task arrives no later than the 100th time slot to ensure successful training. The number of edges K_i for conducting training for each FL task is taken from $[10, 30]$, based on Google data as well. The deadline of each FL task is estimated using $r_{\max}$ and $n_{\max}$, with a linear deadline-violation penalty unless otherwise specified. Without loss of generality, the number of local iterations per global iteration is set to $L_i = 5$; the maximum number of global iterations per time slot is set to $r_{\max} = 9$; the maximum bit precision for quantization is set to $n_{\max} = 32$; and the target accuracy is set to $\varepsilon_i = 0.01$. We set the bidding price for the FL task i from the range $\$[0.1, 1]$.

Cloud-Edge System: We envisage $|\mathcal{N}| = 50$ edges and one cloud. For energy, we adopt $\varphi = 0.005$ [18], $\varphi'_k = 9.15\times10^{-5}$ [33], and $\beta_k = 1.25$ [35]. For system capacity, we set $C = 110$ in proportion to the Google clusters' capacity versus their job arrival trace [26]. For the other parameters, we have $\Upsilon = 1$, $\gamma = 1$, $\sigma_k = 1$, and $\varrho = 0.02$ [32].

Demand Response: We set the energy caps $\{E_t, \forall t\}$ using the real-world EDR events of Elia from September 3, 2019 through September 9, 2019 [29]. We set the electricity price $\{h_t, \forall t\}$ based on the hourly real-time pricing data of ComEd from March 31, 2022 through April 6, 2022 [30].

Algorithms and Implementation: We implement and compare multiple different approaches: (1) the demand response of

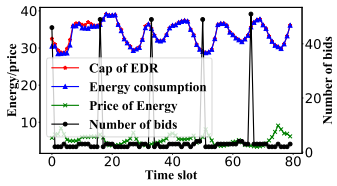

Fig. 3: FL energy consumption

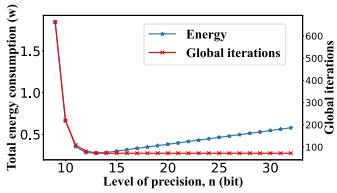

Fig. 4: Energy vs. quantization

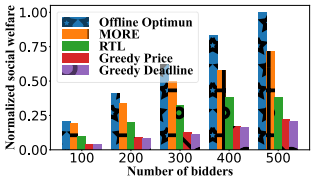

Fig. 5: Social welfare

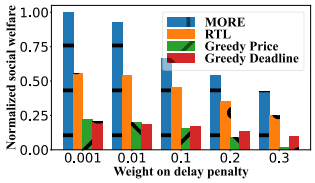

Fig. 6: Deadline violation impact

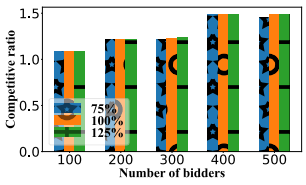

Fig. 7: Empirical competitive ratio

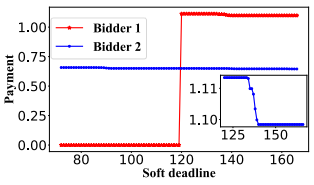

Fig. 8: Payment vs. deadline

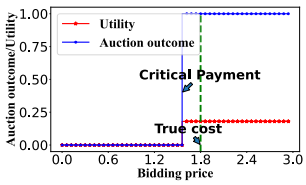

Fig. 9: Truthfulness

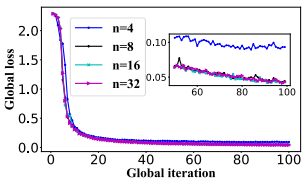

Fig. 10: Loss vs. quantization

federated learning (MORE), which is our proposed approach; (2) "Greedy Price", which, for each time slot, chooses to accept the FL task with the highest bidding price and trains the task in time slots of the lowest energy price until reaching the desired accuracy using $r_{\max}$ and $n_{\max}$; (3) "Greedy Deadline", which, for each time slot, completes training to reach the desired accuracy as soon as possible using $r_{\max}$ and $n_{\max}$; (4) the online algorithm RTL [39]; (5) the offline optimum that solves $\mathbb{P}$ via the Gurobi [40] optimization solver, where all the inputs are assumed known at once in advance.

Our implementation contains about 4,000 lines of Python codes, and we conduct all the evaluations on a desktop with a 2.9-GHz Intel(R) Core(TM) i5 CPU and 8-GB memory.

B. Evaluation Results

Fig. 3 displays the energy cap of EDR, the energy price, the real-time energy consumption of the FL tasks (scheduled by our proposed approach), and the number of the FL tasks arriving at each time slot. The energy price is almost in reverse proportion to the energy cap which is respected at all times.

Fig. 4 exhibits how the FL energy consumption varies as the bit precision for quantization changes for each bid. The number of global iterations and thus the energy consumption decreases as the bit precision for quantization increases. When the bit precision exceeds about 15 bits, the number of global iterations remains basically unchanged; yet, the energy increases since the size of the model becomes larger.

Fig. 5 shows the social welfare comparison for all the implemented algorithms as the number of bids increases. Our approach MORE achieves up to $2\times$ more social welfare than others, and is also closer to the offline optimum. Also, the more bids the system has, the more social welfare it can achieve. This is because the system has more flexibility to choose bids of higher values to optimize the social welfare.

Fig. 6 compares the social welfare of different algorithms as the weight that is associated to the deadline-violation penalty increases. Our approach MORE beats other algorithms, due to explicit consideration of deadlines. Greedy Price performs worse and worse as the weight of deadline violation increases, since it selects FL tasks based on the energy price only. A large weight of deadline violation may decrease the social welfare, since, as the weight grows, our approach tends to reject tasks.

Fig. 7 demonstrates the empirical competitive ratio. We see that, the smaller the value of U/L is, the better the competitive ratio is, aligned with our theoretical analysis. The number of bids has little impact on the competition ratio, and our approach has a relatively stable practical performance. Note that, theoretically, α is also not related to the number of bids.

Fig. 8 depicts the impact of the deadline on the payment. As an example, we select two bids with different bidding prices. Bid 1, initially rejected by our algorithms, is accepted as the deadline extends. Bid 2 pays less payment as the deadline extends. Our algorithms can arrange training schedules more flexibly when the deadline is extended, and tend to arrange schedules at time slots of lower energy prices during EDR.

Fig. 9 confirms truthfulness and individual rationality of our approach. As an example, we pick up one bid. The utility of this bid is maximized when bidding the true valuation. The bidding price, if higher than critical payment, always makes the bid win in the auction. The utility is also non-negative.

Fig. 10 illustrates how the global loss, i.e., the objective to be minimized in the FL tasks, is impacted by the bit precision in quantization. The loss drops as the number of global iterations grows. For running the same number of global iterations, the loss becomes less as the bit precision grows.

TABLE I: Running time

Number of bids	**100**	**200**	**300**	**400**	**500**
Runing time (min)	7.67	14.09	18.51	21.37	24.35

Finally, Table I displays the running time of our approach (excluding the FL training time). Our algorithms finish within 25 minutes in total for 500 tasks in 168 hours. Our algorithms are computationally efficient with acceptable execution time.

VI. Related Work

Cloud/Edge Demand Response: Chen *et al.* [1] propose to switch off entire edge cloudlets to compensate for EDR energy reductions. Cui *et al.* [2] study the mobile edge EDR via game theory. Wang *et al.* [14] focus on the interactions between power grid pricing and data center workload distribution in

EDR. Zhou *et al.* [3] design distributed algorithms across data centers for EDR with auctions. Sun *et al.* [13] use auctions to incentivize tenants' energy reduction in colocations.

These works focus on the EDR of cloud (or data center) and edge systems, yet almost all of them have neglected the AI/ML tasks. None of them have considered the unique features, including model accuracy and quantization, and computing/communication patterns of AI/ML workloads, and thus their solutions are in general inapplicable to AI/ML EDR.

FL Optimization: Luo *et al.* [15] control client selection and local iterations in FL to minimize total cost while ensuring convergence. Vu *et al.* [16] optimize power and computation resources of base stations and clients to save FL energy. Shi *et al.* [17] study bandwidth allocation for devices to attain the best accuracy for FL. Zeng *et al.* [18] jointly control bandwidth, workload partitioning, processor speed of each device to reduce FL energy. Nguyen *et al.* [8] study the optimization of FL clients with computation and communication constraints.

These literatures mainly consider various optimization and control in FL training and/or resource usage individually. They largely ignore quantization, and also the training deadline from a task execution perspective. They do not study the scheduling of many FL tasks, not to mention online EDR and auctions.

AI/ML Task Scheduling: Zhou *et al.* [19] propose a novel multi-task FL framework to enable parallel training. Bao *et al.* [20] present a deep-learning-driven ML cluster scheduler that places training tasks to minimize interference and maximize performance. Zhang *et al.* [21] design an online scheduling method to minimize the average completion time of ML tasks. Shi *et al.* [22] allocate bandwidth and schedule FL to minimize the expected training time with desired accuracy. Sun *et al.* [23] focus on utilization fairness when scheduling ML jobs.

These works study the scheduling of AI/ML tasks, including FL tasks. However, none of them focus on EDR. EDR presents special challenges and requirements for AI/ML workloads, such as auctions and fine-grained energy control via quantization. Thus, such existing research still falls short generally.

VII. Conclusion

Demand response is an important paradigm for operating AI/ML in cloud-edge systems and makes AI/ML, the systems, and the power grid sustainable. While this has been largely ignored in previous studies, our paper aims to bridge the gap. We study the online scheduling of FL tasks that dynamically arrive, while controlling the training and the quantization of each FL task to conform to the demand response energy caps. We design novel reformulation and primal-dual techniques for social welfare optimization, and provably attain multiple performance guarantees, which have also been confirmed by our experimental evaluations. For future research, we will continue to explore the direction of AI/ML sustainability.

Appendix

We prove Theorem 4 in this appendix.

Firstly, we prove that if there exists $\alpha \geq 1$ such that $P_1^i - P_1^{i-1} \geq \frac{1}{\alpha}(D_1^i - D_1^{i-1})$ for every task (or bid) i, then the algorithm is α-competitive. Let P_1^i and D_1^i denote the the value of P_1 and D_1 after processing task i. Obviously, $P_1^{|\mathcal{I}|} = P_1^{|\mathcal{I}|} - P_1^0 = \sum_i (P_1^i - P_1^{i-1})$. With $P_1^i - P_1^{i-1} \geq \frac{1}{\alpha}(D_1^i - D_1^{i-1})$, by summing up these inequalities, we have $\sum_i (P_1^i - P_1^{i-1}) \geq \frac{1}{\alpha}\sum_i (D_1^i - D_1^{i-1}) = \frac{1}{\alpha}(D_1^{|\mathcal{I}|} - D_1^0) = \frac{1}{\alpha} D_1^{|\mathcal{I}|}$. Due to weak duality [24], [36], we have $D_1^{|\mathcal{I}|} \geq P^*$ and thus $P_1^{|\mathcal{I}|} \geq \frac{1}{\alpha} P^*$.

Secondly, we prove that if $m_t^{i-1}(z_t^i - z_t^{i-1}) \geq \frac{1}{\alpha} C(m_t^i - m_t^{i-1}), \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_{i,\hat{l}}$ holds for a given $\alpha \geq 1$, then the algorithm guarantees $P_1^i - P_1^{i-1} \geq \frac{1}{\alpha}(D_1^i - D_1^{i-1})$ for all $i \in \mathcal{I}$. Suppose we interpret m_t^i and e_t^i as the "equipment" cost and the amount of energy to consume from the power grid after processing task i. We use v_t^i to denote the electricity price after processing task i. Note that Algorithm 1 guarantees $P_1^0 = D_1^0 = 0$. When the cloud rejects task i, then $P_1^i - P_1^{i-1} = D_1^i - D_1^{i-1} = 0$. In the next part of the analysis, we assume that task i is accepted and suppose that $\hat{l}$ is the best schedule for it. After processing task i, the increment of the primal objective function (2) is $P_1^i - P_1^{i-1} = b_{i,\hat{l}} - \sum_{t \in \mathcal{T}_{i,\hat{l}}} (f_t(e_t^i) - f_t(e_t^{i-1})) = \mu_i + \sum_{t \in \mathcal{T}_{i,\hat{l}}} \Big(m_t^{i-1} + v_t^{i-1} r_{i,\hat{l}} \Big(\sum_{k \in \mathcal{N}_{i,t}} \big(L_i E_{i,k}(n_{i,\hat{l}}) + 2E'_{i,k}(n_{i,\hat{l}}) \big) + E''_i(n_{i,\hat{l}}) \Big) \Big) - \sum_{t \in \mathcal{T}_{i,\hat{l}}} (f_t(e_t^i) - f_t(e_t^{i-1}))$. Because the left-hand side of Constraint (4) equals the right-hand side when task i is accepted with schedule $\hat{l}$, the second equation is valid. Also, knowing $f_t(e_t) = h_t e_t$, $\sum_{t \in \mathcal{T}_{i,\hat{l}}} (f_t(e_t^i) - f_t(e_t^{i-1})) = \sum_{t \in \mathcal{T}_{i,\hat{l}}} v_t^{i-1} r_{i,\hat{l}} \Big(E''_i(n_{i,\hat{l}}) + \sum_{k \in \mathcal{N}_{i,t}} \big(L_i E_{i,k}(n_{i,\hat{l}}) + 2E'_{i,k}(n_{i,\hat{l}}) \big) \Big)$ and $v_t^{i-1} = h_t$, we have $P_1^i - P_1^{i-1} = \mu_i + \sum_{t \in T_{i,\hat{l}}} m_t^{i-1}$. Thus, according to the definition of v_t, we have $D_1 = \sum_{i \in \hat{l}} \mu_i + \sum_{t \in \mathcal{T}} m_t C$. The dual objective value increases as $D_1^i - D_1^{i-1} = \mu_i + \sum_{t \in \mathcal{T}_{i,\hat{l}}} C(m_t^i - m_t^{i-1})$. Following $m_t^{i-1}(z_t^i - z_t^{i-1}) \geq \frac{1}{\alpha} C(m_t^i - m_t^{i-1})$ and $z_t^i - z_t^{i-1} = 1$, we have $m_t^{i-1} \geq \frac{1}{\alpha} C(m_t^i - m_t^{i-1})$. In addition, when we sum up over all $t \in \mathcal{T}_{i,\hat{l}}$, we can obtain $\sum_{t \in \mathcal{T}_{i,\hat{l}}} m_t^{i-1} \geq \sum_{t \in \mathcal{T}_{i,\hat{l}}} \frac{1}{\alpha} C(m_t^i - m_t^{i-1})$. Obviously, $P_1^i - P_1^{i-1} \geq \mu_i + \frac{1}{\alpha}(D_1^i - D_1^{i-1} - \mu_i)$. Due to $\mu_i \geq 0$ and $\alpha \geq 1$, we have $P_1^i - P_1^{i-1} \geq \frac{1}{\alpha}(D_1^i - D_1^{i-1})$.

Thirdly, we prove that if $m_t dz_t \geq \frac{1}{\alpha} C dm_t, \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_{i,\hat{l}}$ holds for a given $\alpha \geq 1$, then the algorithm guarantees $m_t^{i-1}(z_t^i - z_t^{i-1}) \geq \frac{1}{\alpha} C(m_t^i - m_t^{i-1}), \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_{i,\hat{l}}$. Note that the computation resource consumed by a training task is much less than the cloud capacity in the real world, i.e., $1 \ll C$. We have $dz_t = z_t^i - z_t^{i-1} = 1$. Based on differentiation, we have $dm_t = m_t'(z_t) dz_t = m_t(z_t^i) - m_t(z_t^{i-1}) = m_t^i - m_t^{i-1}$, and $m_t = m_t^{i-1}$ when bid i is submitted. Then, we conclude that $m_t^{i-1}(z_t^i - z_t^{i-1}) \geq \frac{1}{\alpha} C(m_t^i - m_t^{i-1}), \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_{i,\hat{l}}$.

Finally, we prove $\alpha = \max_{t \in \mathcal{T}} \ln(\frac{U - h_t}{L - h_t})$. Following m_t as defined in Equation (5), we have its differential as $dm_t = (L - h_t)(\frac{U - h_t}{L - h_t})^{\frac{z_t}{C}} \ln(\frac{U - h_t}{L - h_t}) \frac{1}{C} dz_t$. Also, note that, for $\alpha_t \geq \ln(\frac{U - h_t}{L - h_t})$, we have $m_t dz_t = (L - h_t)(\frac{U - h_t}{L - h_t})^{\frac{z_t}{C}} dz_t \geq \frac{C}{\alpha_t}(L - h_t)(\frac{U - h_t}{L - h_t})^{\frac{z_t}{C}} \ln(\frac{U - h_t}{L - h_t}) \frac{1}{C} dz_t = \frac{C}{\alpha_t} dm_t$. Therefore, to ensure $m_t dz_t \geq \frac{C}{\alpha} dm_t$, $\forall t$, we can use α to replace α_t, $\forall t$, where $\alpha = \max_{t \in \mathcal{T}} \alpha_t = \max_{t \in \mathcal{T}} \ln(\frac{U - h_t}{L - h_t})$.

REFERENCES

[1] S. Chen, L. Jiao, F. Liu, and L. Wang, "Edgedr: An online mechanism design for demand response in edge clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 2, pp. 343–358, 2022.

[2] G. Cui, Q. He, X. Xia, F. Chen, T. Gu, H. Jin, and Y. Yang, "Demand response in noma-based mobile edge computing: A two-phase game-theoretical approach," *IEEE Transactions on Mobile Computing*, 2021.

[3] Z. Zhou, F. Liu, S. Chen, and Z. Li, "A truthful and efficient incentive mechanism for demand response in green datacenters," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 1, pp. 1–15, 2018.

[4] Y. Yuan, L. Jiao, K. Zhu, and L. Zhang, "Scheduling online ev charging demand response via v2v auctions and local generation," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[5] R. Zhou, Z. Li, C. Wu, and M. Chen, "Demand response in smart grids: A randomized auction approach," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 12, pp. 2540–2553, 2015.

[6] N. C. Luong, P. Wang, D. Niyato, Y.-C. Liang, Z. Han, and F. Hou, "Applications of economic and pricing models for resource management in 5g wireless networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3298–3339, 2018.

[7] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS*, 2017.

[8] N. H. Tran, W. Bao, A. Zomaya, M. N. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *IEEE INFOCOM*, 2019.

[9] C.-J. Wu, D. Brooks, K. Chen, D. Chen, S. Choudhury, M. Dukhan, K. Hazelwood, E. Isaac, Y. Jia, B. Jia *et al.*, "Machine learning at facebook: Understanding inference at the edge," in *IEEE HPCA*, 2019.

[10] H. Wang, Z. Liu, and H. Shen, "Machine learning feature based job scheduling for distributed machine learning clusters," *IEEE/ACM Transactions on Networking*, 2022.

[11] M. Derakhshan, D. M. Pennock, and A. Slivkins, "Beating greedy for approximating reserve prices in multi-unit vcg auctions," in *ACM-SIAM SODA*, 2021.

[12] S. Balseiro, Y. Deng, J. Mao, V. Mirrokni, and S. Zuo, "Robust auction design in the auto-bidding world," in *NeurIPS*, 2021.

[13] Q. Sun, C. Wu, Z. Li, and S. Ren, "Colocation demand response: Joint online mechanisms for individual utility and social welfare maximization," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3978–3992, 2016.

[14] H. Wang, J. Huang, X. Lin, and H. Mohsenian-Rad, "Proactive demand response for data centers: A win-win solution," *IEEE Transactions on Smart Grid*, vol. 7, no. 3, pp. 1584–1596, 2016.

[15] B. Luo, X. Li, S. Wang, J. Huang, and L. Tassiulas, "Cost-effective federated learning design," in *IEEE INFOCOM*, 2021.

[16] T. T. Vu, H. Q. Ngo, D. T. Ngo, M. N. Dao, and E. G. Larsson, "Energy-efficient massive mimo for serving multiple federated learning groups," *arXiv preprint arXiv:2108.13512*, 2021.

[17] W. Shi, S. Zhou, and Z. Niu, "Device scheduling with fast convergence for wireless federated learning," in *IEEE ICC*, 2020.

[18] Q. Zeng, Y. Du, K. Huang, and K. K. Leung, "Energy-efficient resource management for federated edge learning with cpu-gpu heterogeneous computing," *IEEE Transactions on Wireless Communications*, vol. 20, no. 12, pp. 7947–7962, 2021.

[19] C. Zhou, J. Liu, J. Jia, J. Zhou, Y. Zhou, H. Dai, and D. Dou, "Efficient device scheduling with multi-job federated learning," *arXiv preprint arXiv:2112.05928*, 2021.

[20] Y. Bao, Y. Peng, and C. Wu, "Deep learning-based job placement in distributed machine learning clusters," in *IEEE INFOCOM*, 2019.

[21] R. Zhou, J. Pang, Q. Zhang, C. Wu, L. Jiao, Y. Zhong, and Z. Li, "Online scheduling algorithm for heterogeneous distributed machine learning jobs," *IEEE Transactions on Cloud Computing*, 2022.

[22] W. Shi, S. Zhou, and Z. Niu, "Device scheduling with fast convergence for wireless federated learning," in *IEEE ICC*, 2020.

[23] P. Sun, Y. Wen, N. B. Duong Ta, and S. Yan, "Towards distributed machine learning in shared clusters: A dynamically-partitioned approach," in *IEEE SMARTCOMP*, 2017.

[24] N. Buchbinder and J. S. Naor, *The Design of Competitive Online Algorithms via a Primal–Dual Approach*. Now Publishers, Inc., 2009.

[25] R. B. Myerson, "Optimal auction design," *Mathematics of operations research*, vol. 6, no. 1, pp. 58–73, 1981.

[26] "Google cluster data-2011-2." [Online]. Available: https://github.com/google/cluster-data

[27] "Mnist database." [Online]. Available: http://yann.lecun.com/exdb/mnist/

[28] "Cifar-10 dataset." [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[29] "Power generation." [Online]. Available: https://www.elia.be/en/grid-data/data-download-page

[30] "Hourly pricing." [Online]. Available: https://hourlypricing.comed.com/live-prices/

[31] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *ICML*, 2015.

[32] L. Nguyen, P. H. Nguyen, M. Dijk, P. Richtárik, K. Scheinberg, and M. Takác, "Sgd and hogwild! convergence without the bounded gradients assumption," in *ICML*, 2018.

[33] X. Long, J. Wu, and L. Chen, "Energy-efficient offloading in mobile edge computing with edge-cloud collaboration," in *ICA3PP*, 2018.

[34] B. Moons, D. Bankman, and M. Verhelst, *Embedded Deep Learning: Algorithms, Architectures and Circuits for Always-on Neural Network Processing*. Springer, 2019.

[35] B. Moons, K. Goetschalckx, N. Van Berckelaer, and M. Verhelst, "Minimum energy quantized neural networks," in *ACSSC*, 2017.

[36] S. Boyd and L. Vandenberghe, *Convex Optimization*. Convex Optimization, 2004.

[37] Y. Yuan, L. Jiao, K. Zhu, and L. Zhang, "Incentivizing federated learning under long-term energy constraint via online randomized auctions," *IEEE Transactions on Wireless Communications*, vol. 21, no. 7, pp. 5129–5144, 2022.

[38] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[39] N. Jain, I. Menache, J. Naor, and J. Yaniv, "Near-optimal scheduling mechanisms for deadline-sensitive jobs in large computing clusters," *ACM Transactions on Parallel Computing*, vol. 2, no. 1, pp. 1–29, 2015.

[40] "Gurobi." [Online]. Available: http://www.gurobi.com