Budget-Aware Online Control of Edge Federated Learning on Streaming Data with Stochastic Inputs

Yibo Jin, *Student Member, IEEE*, Lei Jiao, *Member, IEEE*, Zhuzhong Qian, *Member, IEEE*, Sheng Zhang, *Member, IEEE*, and Sanglu Lu, *Member, IEEE*

***Abstract*—Performing federated learning continuously in edge networks while training data are dynamically and unpredictably streamed to the devices faces critical challenges, including the global model convergence, the long-term resource budget, and the uncertain stochastic network and execution environment. We formulate an integer program to capture all these challenges, which minimizes the cumulative total latency of stream learning on device and federated learning between devices and the edge server. We then decouple the problem, design an online learning algorithm for controlling the number of local model updates via a convex-concave reformulation and rectified gradient-descent steps, and design a bandit learning algorithm for selecting the edge server for global model aggregations by incorporating the budget information to strike the exploit-explore balance. We rigorously prove the sub-linear regret regarding the optimization objective and the sub-linear constraint violation regarding the maximal on-device load, while guaranteeing the convergence of the global model trained. Extensive evaluations with real-world training data and input traces confirm the empirical superiority of our approach over multiple state-of-the-art algorithms.**

***Index Terms*—Federated Learning, Streaming Data, Budget.**

I. Introduction

Federated learning trains machine learning models via iteratively updating local models on local devices and aggregating the local models from multiple devices to compose the global model on a remote server. This paradigm keeps the training data within each local device and does not upload them to the server [1], thus protecting the users' privacy [2]. However, federated learning often requires all the data samples to be prepared in prior, and then starts the training process which can continuously incur heavy workload (i.e., high CPU load) during the training time period, causing extra waiting time and impacting the use of the devices even with multiplexing.

One approach to avoid such "load congestion" for the local devices is to spread the training load over time, that is, to conduct federated learning as the training data gradually arrive at each local device dynamically. This way, federated learning just runs along with other applications, and therefore does no harm to the normal use of these applications. Unfortunately, this vision of federated learning upon streaming data is non-trivial, especially in edge networks where we need to control both the stream learning on each single device and the edge server selection in a dynamic network environment, as shown in Fig. 1. In fact, we face multiple challenges as follows:

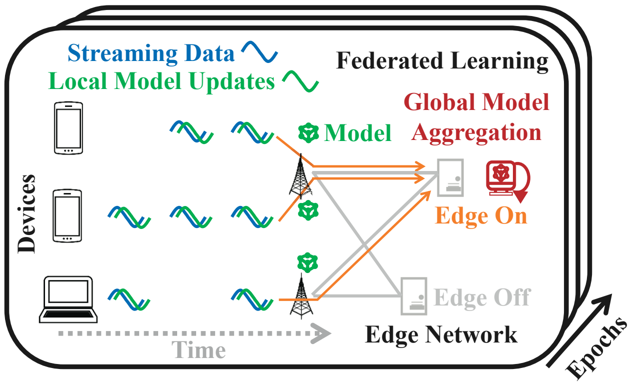


Fig. 1: Edge Federated Learning on Streaming Data

First, learning from streaming data, rather than from static data, in the framework of federated learning complicates the training of the global model. The approach of the Stochastic Gradient Descent (SGD) might be used to dynamically process the streaming data, but it is often hard to converge because not every iteration in SGD can reduce the total loss. So, we would desire a better approach to conduct on-device training upon streaming data, which should consume less computation and avoid continuous high load, and more importantly, we also want to guarantee the global model convergence explicitly.

Second, one may often have a restrictive long-term budget and therefore need to use such budget wisely for renting and using the edge resources for federated learning executions [3]. Inappropriate spending may result in the budget shortage in the future or the unexpected termination of the federated learning process, which can affect the overall training time and the global model convergence. Since we often have no knowledge about the amount of the incoming training data [4], the prices of the edge resources, and other inputs for the future, it is hard to strategically manage the spend of the budget on the fly.

Third, the edge network and execution environment can be intrinsically uncertain and stochastic [5], escalating the difficulty for controlling federated learning in an online manner. For any edge, as the server of the global aggregation, both the network delay of transferring models and the execution delay of performing the aggregations can vary as time goes [6]—we

Manuscript received February 28, 2021; revised July 25, 2021 and September 13, 2021; accepted September 22, 2021. Date of publication XXX XX, 2021; date of current version XXX XX, 2021. This work was supported in part by the National Key R&D Program of China under Grant No. 2017YFB1001801, in part by the National Natural Science Foundation of China under Grant No. 61832005 and No. 61872175, in part by the Natural Science Foundation of Jiangsu Province under Grant No. BK20181252, and in part by the U.S. National Science Foundation under Grant CNS-2047719. *(Corresponding authors: Lei Jiao; Zhuzhong Qian; Sheng Zhang.)*

Y. Jin, Z. Qian, S. Zhang, and S. Lu are with the State Key Laboratory for Novel Software Technology, the Department of Computer Science and Technology, Nanjing University, Nanjing 210023, China (e-mail: yibo.jin@smail.nju.edu.cn, {qzz, sheng, sanglu}@nju.edu.cn).

L. Jiao is with the Department of Computer and Information Science, University of Oregon, Eugene, OR 97403, USA (e-mail: jiao@cs.uoregon.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier xx.xxxx/JSAC.2021.xxxxxxx

need to dynamically strike the balance between exploiting the best edge so far for which we have the historical observations of its performance and exploring a new but possibly better edge for which we have no information yet. While the inputs can come from unknown stochastic distributions and are not observable beforehand, we seek to make irrevocable decisions online to optimize the expected performance in the long run.

Existing research falls insufficient for addressing the aforementioned challenges. Some focus on training over streaming data [7–11], but none of them are for distributed federated learning. Others have considered the optimization of federated learning [12–16] at network edges, but they largely neglect the dynamic and unpredictable arrivals of training data. It is desired that the features of streaming data and the convergence of federated learning are jointly considered and well addressed.

We firstly model the target scenario of controlling federated learning upon streaming data in a long-term scope, as shown in Fig. 1. Our model considers the overall latency incurred by both computation and communication of local model updates in on-device stream learning and global aggregations at the edge server in federated learning over the entire time horizon. Our model features edge resource restrictions, global model convergence, and the long-term budget under unpredictable data arrivals, dynamic resource prices, and heterogeneous and stochastic edge performance. Upon the models, we formulate our control problem for optimization correspondingly.

We then propose and design an online approach to solve this problem through decomposing it into two subproblems and solving them separately in every time epoch. The key of this decomposition is to separate the intertwined control decisions and related delays they incur using the estimate of the lower bound of the number of time epochs for which federated learning can be conducted under the given budget. For the first subproblem of determining the number of local model updates in stream learning, we design an online learning algorithm via a convex-concave reformulation and rectified gradient-decent-based steps without relying on further future inputs. For the second subproblem of selecting the edge server for model aggregation, we design a bandit learning algorithm via carefully incorporating the budget information into the upper confidence bound for each edge and using this bound to maintain the exploit-explore balance across different edges.

Further, we perform rigorous theoretical analysis of our proposed online algorithms. We prove that, for our optimization objective of the cumulative total latency, the "regret" in terms of the expected difference between the latency incurred by our online control decisions and that incurred by the offline optimal control decisions has a sub-linear upper bound in terms of the stopping time for a given budget. We also prove that, for our constraints, the maximal on-device load incurred by stream learning is upper-bounded sub-linearly with regards to the stopping time, while the global model convergence is ensured. Note that such results are non-trivial, which require all our proposed algorithms to work together and differ our work from most existing analysis for online algorithms.

Finally, we conduct extensive experiments using real-world training data and input traces, including the training datasets of a9a [17], rcv [18], movielens100k [19], and movielens1m [19],

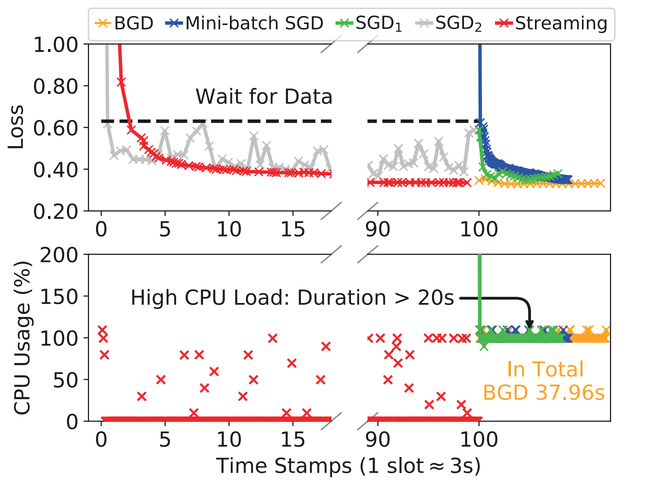


Fig. 2: Comparison of Different Training Algorithms

TABLE I: Cumulative Training Delay (s)

Dataset	Mini-batch SGD	Streaming	Loss Difference
a9a [17]	30.49	14.08	0.00
rcv [18]	74.35	49.18	0.05
movielens100k [19]	33.78	27.72	0.06
movielens1m [19]	324.41	277.82	0.07

the location and the bandwidth data of edges [20], the round-trip times between the users and edges [21], as well as the dynamic prices regarding renting the edges [3]. We compare our approach to multiple alternative approaches under a variety of settings, and find the following results: i) for different training datasets streamed, our proposed approach reduces the peak latency across training epochs to no more than 3 seconds, and cumulatively consumes only several minutes on training over the 74-minute-long time horizon of data streaming, with a very moderate 0.07 total loss degradation; ii) compared to alternative approaches with different combinations of on-device local training algorithms and edge selection strategies, our approach consistently performs the best in terms of cumulative latency and training loss; iii) our approach scales well in latency and loss, and grows slowly as the budget becomes less, the training data arrive faster, the number of devices becomes larger, the size of the trained model becomes larger, and the computing power of devices becomes weaker.

II. Motivation, Modeling, and Formulation

A. Background and Motivation

We first show our preliminary case studies on the difference between learning on static data and learning on streaming data.

Learning upon Static Data: Conventionally, model training requires all the data samples to be ready in prior. Many gradient-based approaches [9], such as Batch Gradient Decent (BGD) and Stochastic Gradient Decent (SGD), are adopted in this case. Regarding using the gradient to iteratively update the model parameters, BGD calculates the gradient over the entire dataset in each iteration (i.e., upon the average of the gradients over all data samples), while SGD calculates the gradient over a single data sample in each iteration. Therefore, BGD uses the "real" gradient of the total loss to be minimized, while SGD can be regarded as using an approximation of the real gradient

TABLE II: Summary of Major Notations

Inputs	Descriptions
n_{it}	[1]Volume of data arrived at device i in epoch t
h_{ijt}	Delay incurred by local model update per iteration, which is relevant to arrived data volume, $h_{it} = \sum_j h_{ijt}$
a_{iet}	[1]Round-trip time from device i to edge e in epoch t
b_{iet}	[1]Bandwidth from device i to edge e in epoch t
c_{et}	[1]Execution time of edge e for model aggregation in epoch t
r_{et}	Renting cost of edge e in epoch t
ε	Desired loss achieved by federated learning
π	Maximal resource used on devices in an epoch
τ_B	Time stamp of the stopping time, with respect to budget B
Decisions[2]	**Descriptions**
ρ_t	Local iterations for local model updates
x_{et}	Whether edge e is decided to be the location for global model aggregation at epoch t or not

1. Those inputs are posterior; $\{a_{iet}\}, \{b_{iet}\}, \{c_{et}\}$ are further stochastic.
2. $\{\bar{\rho}_t\}$ and $\{\bar{x}_{et}\}$ are the results produced by our designed algorithm.

but often consumes less time for related gradient computation. As a trade-off, the mini-batch SGD approach [1, 22] splits the training data into multiple subsets, and then calculates the gradient over a subset in each iteration.

Learning upon Streaming Data: Different from the previous setting, where all training data are available before the training process starts, learning over streaming data refers to the new setting, where the training data arrive dynamically on the fly, as the training process goes. The training approaches in this category include SAGA [9] and strSAGA [11]. These approaches are derived by adapting SGD, where within each iteration, the gradient is firstly calculated upon the new data samples that have just arrived and then revised by the data that arrived previously before the current data samples. Specifically, such a revision of the gradient per iteration is calculated upon the computation of a well-designed weighted sum [14], whose weights are maintained for all those data samples that have arrived. Unlike BGD and mini-batch SGD, this approach amortizes the computation for the online scenario, and related calculation of the designated weighted sum upon the weights of the data samples actually controls the decent steps of the gradients, leading to a faster model convergence.

Case Study: As in Fig. 2, we compare multiple approaches: SGD upon static data (denoted as SGD$_1$), mini-batch SGD upon static data, BGD upon static data, SGD upon streaming data (denoted as SGD$_2$), and strSAGA upon streaming data.

For static data, all the data are prepared in prior, and related SGD approach (SGD$_1$) processes all data samples one after another; because all the data are ready in this case, the SGD approach can pass all the data multiple times (e.g., from the first data sample to the last one and then from the first to the last again) to reduce the loss for convergence. In Fig. 2, SGD$_1$ passes all the data 20 times. In contrast, for streaming data, each data sample dynamically arrives, and the SGD approach (SGD$_2$) passes each data sample once and only once.

As data are streamed, the first three approaches wait for 300s until all the data arrive before starting to train the model, and the last two approaches directly starts as data begin to arrive. Regarding the loss, we find that, to reduce the total loss to approximately the same level, strSAGA finishes earliest; SGD$_1$ is efficient, and it completes with the shortest time duration after all data become available; BGD reaches the best loss,

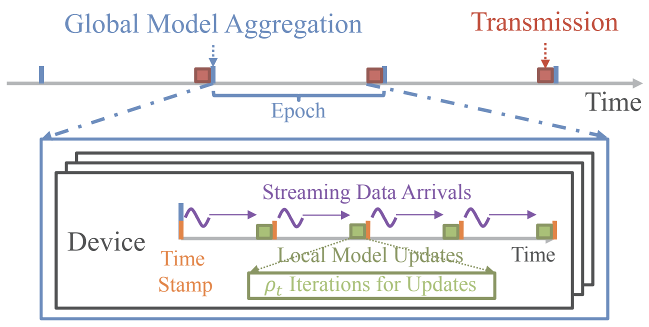


Fig. 3: Illustration of Federated Learning upon Streaming Data

but takes 38s to complete; and the loss of SGD$_2$ fluctuates, since the decent step of the gradient calculated only takes one data sample into consideration. Regarding CPU usage, the first three approaches continuously and fully occupy one CPU core in our multi-core system; strSAGA distributes the workload over time and keeps pretty low CPU usage most of the time.

The data samples dynamically and continuously arrive one after another per time slot, and the stream training algorithm is executed in real time—this does not mean the stream training algorithm uses up all the 3s of each time slot for execution; instead, it only uses hundreds of milliseconds in a time slot for execution, because after processing the model updates, it "suspends" and waits until the trigger of the next time slot and then "resumes" to process the new data samples. Therefore, the total amount of time for training algorithm execution, rather than waiting, across all of the 100 time slots is 14.08s. This is also verified in the bottom figure of Fig. 2: only when the training is actually conducted, the CPU usage becomes high (implied by the red cross signs) in the streaming case.

The settings of our experiments are as follows. Data arrivals obey a Poisson process, lasting for 5 minutes. The dataset [17] used is a9a. Additional results on various datasets [18, 19] are shown in TABLE I. Mini-batch SGD processes 50 data samples per batch, and 20 local rounds used for these approaches except for strSAGA according to the previous work [1]. Each local round passes all the batches once. The step size used for the gradient updates is 2e-1. We implement and conduct our experiments based on the previous work [11], via Python 2.7 upon Ubuntu 16.04 with 4 CPU cores and 8 GB memory.

These results motivate us to study stream learning. In this paper, we consider a more comprehensive setting of distributed *federated learning* over streaming data upon strSAGA, optimizing training time while ensuring model convergence.

B. System Settings and Models

We summarize all the major notations in TABLE II.

Edge Infrastructure: We study the system over a series of epochs. Within each epoch $t \in \mathcal{T}$, the volume of data that arrive on device i at time stamp $j \in \mathcal{M}_t$ is v_{ijt}, where $i \in \mathcal{N}$ is the index of the device and $\mathcal{M}_t$ is the set of time stamps in epoch t. We denote by $n_{it} = \sum_{j \in \mathcal{M}_t} v_{ijt}$ the total volume of all streaming data that arrive at device i in epoch t. We then consider a service provider that rents a set of distributed edges $\mathcal{E}$, where an "edge" here refers to a micro data center or a server cluster, co-located at a WiFi access point or a cellular

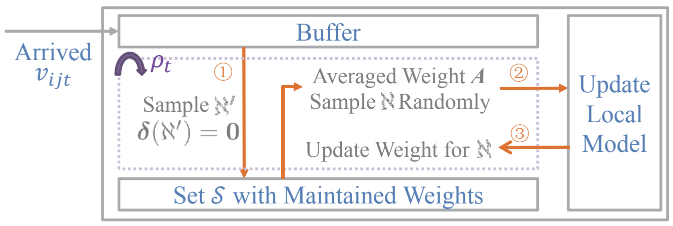


Fig. 4: Illustration of Local Training for Streaming Data

base station. Within each epoch t, the bandwidth and round-trip time between device i and edge e are represented as b_{iet} and a_{iet}, respectively; and the cost for renting edge e is r_{et}.

We study the system over a time horizon which consists of a series of consecutive "epochs", shown in Fig. 3. During each epoch, the training data arrive sequentially in mini-batches at different time stamps. For each of such mini-batches, we run ρ_t iterations to update the local model on the device, where in each iteration we pick up one data sample randomly, calculate the gradient and related average weights, and update the local model and the weight of the selected data sample. Only when finishing processing the last mini-batch of data in the epoch, we do an aggregation (i.e., sending the local model of each device to a common selected edge to produce the global model following the conventional federated learning paradigm), and then we enter the next epoch and send the latest global model back to each device as the local model to be updated.

We are minimizing the overall latency (i.e., the computation time (for local model training and global model aggregation) on all the devices plus the transmission time (for sending the local models and retrieving the global model) between the devices and the edge). We consider such time consumption as the computation and communication overhead of the system.

Different from traditional federated learning, we do one and only one aggregation at (the end of) each epoch (before the stopping epoch subject to the budget constraint).

Stream Learning on Device: We adopt strSAGA (shown in detail in the algorithm) to train the local model on each device which has a separate stream of arriving data. On device i in epoch t, strSAGA consists of multiple local iterations, where each iteration z further consists of three steps, shown in Fig. 4:

i) The data that have just arrived are maintained by a buffer. Then, randomly move a data sample $\aleph'$ from the buffer (if the buffer has data samples) to a set $\boldsymbol{\mathcal{S}}$ with initialized "weight" $\mathbf{0}$ (the weight is actually a vector, denoted by $\boldsymbol{\delta}(\aleph')=\mathbf{0}$); When a data sample with initialized weight is moved to $\boldsymbol{\mathcal{S}}$, we use a vector $\boldsymbol{A}$ to measure the latest averaged weight of $\boldsymbol{\mathcal{S}}$ as

$$\boldsymbol{A}=\sum_{s\in\boldsymbol{\mathcal{S}}}\boldsymbol{\delta}(s)/|\boldsymbol{\mathcal{S}}|, \tag{0a}$$

where the set $\boldsymbol{\mathcal{S}}$ maintains the data that have arrived and are the candidates for model updates; Here, $|\boldsymbol{\mathcal{S}}|$ indicates the number of the elements maintained by $\boldsymbol{\mathcal{S}}$; The total number of the data samples that arrive at time stamp j is v_{ijt};

ii) Randomly choose one data sample $\aleph$ from $\boldsymbol{\mathcal{S}}$, where such data sample is sampled from the whole set, and may not be the data sample $\aleph'$ just moved from the buffer in previous step;

iii) Compute the gradient $\boldsymbol{\sigma}$ for $\aleph$ upon the loss function $\mathcal{L}$, and the model $\tilde{\boldsymbol{w}}_{z-1}$ obtained from previous iteration:

$$\boldsymbol{\sigma}=\nabla\mathcal{L}(\aleph,\tilde{\boldsymbol{w}}_{z-1}), \tag{0b}$$

where ∇ is the notation of the gradient of $\mathcal{L}$ with respect to $\aleph$; Note that the index of current iteration is z; Then, update the model upon $\boldsymbol{\sigma}$ just calculated as follows:

$$\tilde{\boldsymbol{w}}_z=\tilde{\boldsymbol{w}}_{z-1}-\eta(\boldsymbol{\sigma}-\boldsymbol{\delta}(\aleph)+\boldsymbol{A}), \tag{0c}$$

where $\tilde{\boldsymbol{w}}_z$ is the latest model; Note that, when updating the model, the "old" weight of $\aleph$ maintained by $\boldsymbol{\mathcal{S}}$ is used (i.e., $\boldsymbol{\delta}(\aleph)$); After the model update, the weight of $\aleph$ is updated (i.e., $\boldsymbol{\delta}(\aleph)\leftarrow\boldsymbol{\sigma}$). η is a step size and $\aleph$ is still maintained by $\boldsymbol{\mathcal{S}}$.

In Step 1, we move one data sample into the set $\boldsymbol{\mathcal{S}}$; in Step 2, we randomly sample one data sample from the set $\boldsymbol{\mathcal{S}}$. Thus, the data sample that is moved in Step 1 is indeed not necessarily the data sample that is sampled in Step 2. We use different symbols for Step 1 ($\aleph'$) and Step 2 ($\aleph$). Regarding Step 3, we calculate the gradient and also use this gradient to update the model. These three steps are a skeleton of Algorithm 2.

This algorithm (details shown in algorithm section) achieves the following model convergence mentioned in [11]:

$$E[l_{it}^U-l_{it}^*]\le u\cdot\max\{1,(\frac{2n_{it}}{M_t\rho_t})^\alpha\}\cdot\mathcal{H}_{it}(n_{it}), \tag{0d}$$

where l_{it}^U is the loss incurred by the model updated, following the above iterative algorithm; l_{it}^* is the best loss (which is often unknown) achieved by the oracle; ρ_{it} is the number of local iterations; $\mathcal{H}_{it}(n_{it})$ is the loss obtained by the "empirical risk minimizer"; and u and α are constants, where $1/2<\alpha<1$. While $E[l_{it}^U-l_{it}^*]$ refers to the difference of the loss over the entire data distribution (assumed independent and identically distributed), $\mathcal{H}_{it}(n_{it})$ here refers to the empirical minimum loss over the data samples that actually arrive in the system.

$$l_{it}^U=\frac{1}{n_{it}}\sum_{\aleph\in\boldsymbol{\mathcal{D}_{it}}}\mathcal{L}(\aleph,\tilde{\boldsymbol{w}}_{it}^U),\quad l_{it}^*=\frac{1}{n_{it}}\sum_{\aleph\in\boldsymbol{\mathcal{D}_{it}}}\mathcal{L}(\aleph,\tilde{\boldsymbol{w}}_{it}^*), \tag{0e}$$

where $\boldsymbol{\mathcal{D}_{it}}$ is the set of all data arriving in epoch t on device i; $n_{it}=|\boldsymbol{\mathcal{D}_{it}}|$; $\tilde{\boldsymbol{w}}_{it}^U$ is the model obtained via the stream training at the end of epoch t; $\tilde{\boldsymbol{w}}_{it}^*$ is the optimal model in epoch t on device i, which is defined as $\arg\min_{\tilde{\boldsymbol{w}}}\frac{1}{n_{it}}\sum_{\aleph\in\boldsymbol{\mathcal{D}_{it}}}\mathcal{L}(\aleph,\tilde{\boldsymbol{w}})$.

"$v_{ijt}<\rho_t$" does not prevent the execution. ρ_t refers to the number of data samples sampled from the "base" (i.e., the set $\boldsymbol{\mathcal{S}}$). Note that we do not require the set $\boldsymbol{\mathcal{S}}$ to contain more than ρ_t data samples; if $\boldsymbol{\mathcal{S}}$ contains fewer than ρ_t data samples, we can still do the sampling for ρ_t times even if in this case some data samples will be selected more than once. We indeed allow taking repeated data samples. Also, our theoretical analysis does not assume "$v_{ijt}<\rho_t$" or "$v_{ijt}\ge\rho_t$". Therefore, our current analysis still works when "$v_{ijt}<\rho_t$".

Federated Learning across Device and Edge: In federated learning, in each epoch the local model on each device is firstly updated by the local model updates using the streaming data, and then at the end of the epoch, each local model is sent to the edge for aggregation in order to generate the global model [1]. The global model is then sent back to every participating device for the local model updates in the next epoch.

We focus on the overall latency incurred by the federated learning. We denote by h_{ijt} the computation time consumed by the local model updates on device i over data arriving at time stamp j in epoch t. As the amount of computation is proportional to the data volume and the number of local iterations, the overall computation time over the streaming data on device i in epoch t is $h_{it}\rho_t$, where $h_{it} = \sum_j h_{ijt}$.

The amount of computation is proportional to the number of "iterations" ρ_t. Since we only process one single data sample per iteration, the amount of computation can also be viewed as proportional to the size of the data that we process.

When conducting the global model aggregation, we consider the transmission delay and the propagation delay as

$$\beta_{iet} = 2w/b_{iet} + a_{iet}, \tag{0f}$$

where β_{iet} is the overall transmission delay from device i to edge e in epoch t; b_{iet} is the bandwidth; w is the model size; a_{iet} is the propagation time. $2w/b_{iet}$ covers both trips from device to edge and from edge to device. We are also aware that the available bandwidth could be asymmetric (i.e., the bandwidth from edge to device is different from the bandwidth from device to edge). In this case, our notation b_{iet} can refer to the bottleneck bandwidth between the two values during the transmissions. The execution time consumed by the global model aggregation at edge e in epoch t is c_{et}.

We emphasize that $\{a_{iet}\}$, $\{b_{iet}\}$, and $\{c_{et}\}$ are all stochastic inputs (i.e., in every epoch t, we only observe a sample of each of these three inputs for each device and selected edge).

The transmission time is not always a stable value and could vary dynamically from time to time in reality. Such dynamism and uncertainty is a norm and very typical, especially when the device is connecting to the edge via cellular wireless networks. In order to capture this dynamism and uncertainty, we model both the (available) bandwidth and the propagation delay as inputs sampled from stochastic distributions, which aligns with lots of existing research, including [6, 23, 24]. By "stochastic", we mean that the transmission time between a device and an edge in the current epoch would likely be different from the transmission time between this same device and this same edge in a future epoch; in each of these two epochs, we could only observe a sample value of the transmission time from the underlying distribution, and the two sample values of the transmission time for the two epochs could be different.

a_{iet} is the value observed and sampled at the epoch t from the stochastic distribution of the round-trip propagation time between the device i and the edge e; b_{iet} is the value observed or sampled at the epoch t from the stochastic distribution of the (available) bandwidth between the device i and the edge e; c_{et} is the value observed and sampled at the epoch t from the stochastic distribution of the computation time for executing the global aggregation performed on the edge e.

Control Decisions: We make two types of control decisions. The first is ρ_t (i.e., the (integral) number of iterations for the local model updates on each device within epoch t). Although the streaming learning on device calculates the gradient using one data sample per time, such local model update may have multiple iterations. The next is x_{et} (i.e., the (binary) decision of whether or not edge e is used as the location for the model aggregation within epoch t). After completing the local model updates on each device, we have the flexibility to select one edge for executing the global model aggregation.

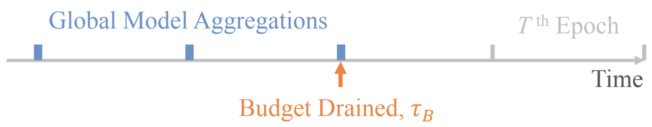


Fig. 5: Illustration of Stopping Time

C. Problem Formulation and Algorithmic Challenges

Control Problem $\mathbb{P}$: With the system models shown above, we formulate the following optimization problem to control the federated learning process over the streaming data:

$$\min \quad \mathcal{P} = \sum_{t\le\tau_B} \{ \sum_i \{h_{it}\rho_t + \sum_e x_{et}\beta_{iet}\} + \sum_e c_{et}x_{et} \}$$

$$s.t. \quad \max_j\{\rho_t h_{ijt}\} \le \rho_t h_{it} \le \pi, \ \forall t \le \tau_B, \forall i, \tag{1}$$

$$\sum_e x_{et} = 1, \ \forall t \le \tau_B, \quad \sum_{t\le\tau_B}\sum_e r_{et}x_{et} \le B, \tag{2}$$

$$\sum_{t\le\tau_B}\sum_e x_{et} \ge \mathcal{O}(\sum_{t\le\tau_B} \rho_t^{-\alpha}/\varepsilon), \tag{3}$$

$$var. \quad x_{et} \in \{1, 0\}, \rho_t \in \mathbb{Z}^+. \tag{4}$$

The objective is to minimize the overall latency of the whole lifecycle of federated learning upon streaming data, including local training time, model transmission and propagation time, and global aggregation time, where τ_B is the time stamp of stopping, as the entire training process terminates due to the budget B. Here, Constraint (1) ensures that the maximal training load incurred on devices by the streaming data is controlled within an acceptable threshold π. Constraint (2) ensures that within each epoch, only one edge is selected as the location for global model aggregation and the cumulative cost of renting the edges cannot exceed the budget B. Constraint (3) ensures global model convergence according to the lemma shown later. Constraint (4) specifies the variables' domains.

The value of $\sum_{t\le\tau_B,e} x_{et}$ is actually τ_B, and therefore the entire left-hand side of the Constraint (3) can then become τ_B; however, x_{et} indeed has the impact, because actually we have $\tau_B = \max_\tau\{\tau | \sum_{t\le\tau}\sum_e r_{et}x_{et} \le B\}$, where $\sum_e x_{et} = 1$.

We believe there are two ways to interpret this. First, we say x_{et} is a decision variable, and τ_B is then a function of the decision variable x_{et} (as it is a function, τ_B is not a decision variable). Second, we say x_{et} is a decision variable and τ_B is another decision variable; besides, in this case, we need to add the equation $\tau_B = \max_\tau\{\tau | \sum_{t\le\tau}\sum_e r_{et}x_{et} \le B\}$ as a new constraint to our current problem formulation. Both the first approach and the second approach here can have τ_B in the objection function and make sense. As illustrated in Fig. 5, the physical meaning of τ_B is the "stopping time", and the optimization objective considers the time scope of $t \le \tau_B$, where $\tau_B \le |\mathcal{T}|$, rather than considering the time scope of $t \le |\mathcal{T}|$, where $\mathcal{T}$ is the set of all (possible) epochs.

All the devices under our consideration need to participate in global model aggregation in this paper. We have adopted the overall transmission delay as part of the optimization objective (i.e., $\sum_{t\le\tau_B}\sum_e x_{et}\beta_{iet}$) in this paper. Using the maximum transmission delay to replace the overall transmission delay is a valid alternative—both approaches have been adopted in various existing research. As we intended to use transmission delay to reflect the communication overhead, we chose to use the total delay (i.e., the total overhead) instead of the maximum delay (i.e., the maximum overhead) in the objective.

The term used in $\mathcal{O}$ of Constraint (3) is shown in Lemma 1:

Lemma 1. *If the number of global aggregations $\sum_{t\le\tau_B,e} x_{et}$ reaches $\tau_B \ge \mathcal{O}(\sum_{t\le\tau_B}\rho_t^{-\alpha}/\varepsilon)$, the global model convergence is ensured (i.e., the following inequality holds):*

$$E[\frac{\sum_t l_t}{\tau_B} - l_{opt}] \le \frac{\sum_{i,t} u \max\{1, (\frac{2n_{it}}{M_t\rho_t})^{\alpha}\}\mathcal{H}_{max}}{N\tau_B} \le \varepsilon,$$

where $\mathcal{H}_{max} = \max\{\mathcal{H}_{it}\}$; ε is the desired global loss; N is the number of devices (i.e., $|\boldsymbol{\mathcal{N}}|$); l_t is the loss of the global model that is produced by federated learning with strSAGA; and l_{opt} is the optimal loss of the global model.

Proof. See Appendix A. Based on Inequality (0d). □

ε here, as the desired (upper bound of the) global loss to be achieved, needs to be set to make sense. For instance, we expect it to be a small value. Aligned with previous works [11], we set ε upon the loss obtained in the case of static data and non-distributed training (e.g., 0.01 $\sim$ 0.001 for "a9a").

l_{opt} is the loss over all data via the optimum model from federated learning: $l_{opt} = \frac{1}{N\tau_B}\sum_{i,t}\frac{1}{n_{it}}\sum_{\aleph\in\boldsymbol{\mathcal{D}_{it}}}\mathcal{L}(\aleph, \tilde{\boldsymbol{w}}^*_{FL})$, where $\boldsymbol{\mathcal{D}_{it}}$ is the set of all data arrived in epoch t on device i, and $n_{it} = |\boldsymbol{\mathcal{D}_{it}}|$; $\tilde{\boldsymbol{w}}^*_{FL}$ is the optimum model obtained by federated learning; $\mathcal{L}$ is the loss function; N is the number of devices; and τ_B is the stopping time in Fig. 5.

Algorithmic Goal: In this paper, *our goal is to design an algorithm which, in an online manner, produces $\bar{\rho}_t$ and $\bar{x}_{et}$ to solve the problem $\mathbb{P}$ while upper-bounding the regret as $E_{\xi\sim\mathscr{D}}[\bar{\mathcal{P}}_\xi - \mathcal{P}^*_\xi] \le C$.* Here, denoting $\xi = \{a_{iet}, b_{iet}, c_{et}\}$ as the specific inputs sampled from the unknown distribution $\mathscr{D}$, we define $\bar{\mathcal{P}}_\xi = \mathcal{P}_\xi(\bar{\rho}_t, \bar{x}_{et})$ and thus use $E_{\xi\sim\mathscr{D}}[\bar{\mathcal{P}}_\xi]$ to refer to the expected objective (i.e., latency) using the decisions $\bar{\rho}_t$ and $\bar{x}_{et}$ generated by our approach to be designed; we also define $\mathcal{P}^*_\xi = \mathcal{P}_\xi(\rho^*_t, x^*_{et})$, where ρ^*_t and x^*_{et} are the optimal decisions under ξ, and thus use $E_{\xi\sim\mathscr{D}}[\mathcal{P}^*_\xi]$ to represent the expected optimum; finally, we define $E_{\xi\sim\mathscr{D}}[\bar{\mathcal{P}}_\xi - \mathcal{P}^*_\xi]$ as the "regret", and C is a constant in terms of the given budget.

Algorithmic Challenges: Solving this problem to achieve the above goal is non-trivial, due to the following challenges:

Stochastic Uncertainty: The inputs $\{a_{iet}\}$, $\{b_{iet}\}$, and $\{c_{et}\}$ are stochastic. In every epoch t, we only observe the samples of these inputs from their unknown distribution and need to make the control decisions online based on such sampled inputs. Note that the regret is defined in the sense of expectation. Thus, the control algorithm we design needs to work well not only for the observed samples of the inputs but also for those samples that are not observable—it needs to work well for the entire distribution which is yet unknown. Even though other inputs such as $\{v_{ijt}, n_{it}\}$ are not stochastic, these stochastic inputs have escalated the difficulty for the algorithm design.

For the edge we select, the observed values of a, b, and c are sample values from their corresponding stochastic distributions which are unknown to us. If we choose the same edge again in a future time epoch, the values of a, b, and c will likely be different, because they can be different sample values. Yet, we highlight that our job in this paper is to design algorithms that work well in the *expectation* sense for these stochastic inputs based on those specific sample values of these inputs that we observe as we run our algorithms on the fly.

Long-Term Budget: The problem is equipped with the long-term constraints (2) and (3), and the length of the entire time horizon (i.e., total number of epochs) depends on the budget B. Even with all the inputs observed as time goes, it is hard to manage the use of the constrained budget on the fly while pursuing the minimization of our proposed objective.

Intractability: The proposed problem has a linear objective, with non-linear terms, such as the maximal load and the model convergence, in the constraints. Even making such non-linear terms linear and removing the stochasticity, proposed problem is still hard to solve in an offline setting, not to mention that we desire to solve it in an online manner.

III. Online Algorithm Design

In order to solve the problem proposed in an online manner. We try to decouple the problem into two subproblems. Unfortunately, the subproblems decoupled are still challengeable due to long-term guarantee, integer domain and stochastic inputs.

The first subproblem is further split into a series of one-shot minimization problems for each epoch, in order to decide the local iterations based on novel online learning technique. The second subproblem is then solved by using bandit technique per epoch, to decide the edge location for model aggregation. The relationships of these subproblems are shown in Fig. 6.

More specifically, Algorithm 1 is used as the structure of our proposed online schema. Algorithm 2 is used for controlling the local model updates on streaming data while Algorithm 3 is used for controlling the global model aggregations.

A. Problem Decomposition

To facilitate the design of our online algorithm, we introduce some additional notations and auxiliary subproblems:

$$\min \sum_{t\le\tau_B} f_t(\rho_t) \triangleq \sum_{t\le\tau_B}\sum_i h_{it}\rho_t \qquad [\mathbb{P}_1]$$

$$s.t. \sum\nolimits_{t\le\tau_B} g_{1t}(\rho_t) \triangleq \sum\nolimits_{t\le\tau_B}\{\rho_t h_{it} - \pi\} \le 0, \qquad (5)$$

$$\sum\nolimits_{t\le\tau_B} g_{2t}(\rho_t) \triangleq \sum_{t\le\tau_B}\{\frac{\varpi_t\rho_t^{-\alpha}}{\varepsilon} - \frac{\Xi_1(B)}{\Xi_2(B)}\} \le 0, \quad (6)$$

$$var. \ \rho_t \in \mathbb{Z}^+,$$

where all of these terms regarding ρ_t is decomposed from $\mathbb{P}$, and Constraint (5) and Constraint (6) are the long-term version of Constraint (1) and (3), respectively. Within Constraint (6), $\Xi_1(B)$ and $\Xi_2(B)$ are defined as follows upon [25]:

$$\Xi_1(B) \triangleq \varrho_0 B - \varrho_2 - \varrho_3 log(\varrho_1 + \varrho_0 B),$$
$$\Xi_2(B) \triangleq 2\varrho_0 B + \varrho_1, \qquad (7)$$

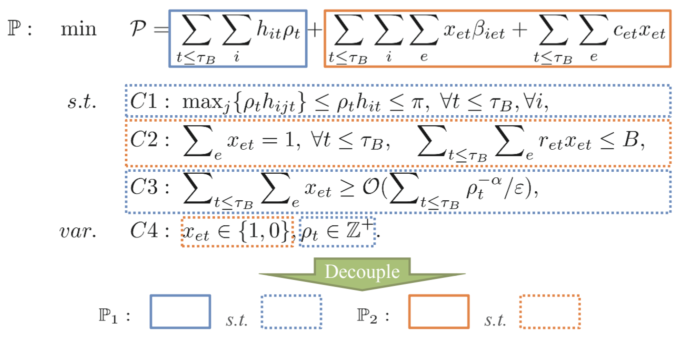

$\mathbb{P}$: min $\mathcal{P} = \sum_{t\le\tau_B}\sum_i h_{it}\rho_t + \sum_{t\le\tau_B}\sum_i\sum_e x_{et}\beta_{iet} + \sum_{t\le\tau_B}\sum_e c_{et}x_{et}$
s.t. $C1: \max_j\{\rho_t h_{ijt}\} \le \rho_t h_{it} \le \pi,\ \forall t \le \tau_B, \forall i,$
$C2: \sum_e x_{et} = 1,\ \forall t \le \tau_B,\quad \sum_{t\le\tau_B}\sum_e r_{et}x_{et} \le B,$
$C3: \sum_{t\le\tau_B}\sum_e x_{et} \ge \mathcal{O}(\sum_{t\le\tau_B}\rho_t^{-\alpha}/\varepsilon),$
var. $C4: x_{et} \in \{1,0\}; \rho_t \in \mathbb{Z}^+.$
Decouple
$\mathbb{P}_1$: s.t.
$\mathbb{P}_2$: s.t.

Fig. 6: Relationship between Proposed Problems

where ϱ_0 to ϱ_3 and ϖ_t are all constants. When decoupling $\mathbb{P}$ into subproblems and making further transformations to solve $\mathbb{P}_1$, we have actually replaced big-$\mathcal{O}$ with specific parameters. $\varpi_t = \sum_i u \max\{1, (\frac{2n_{it}}{M_t})^\alpha \mathcal{H}_{max}\}/N$. Here, $\varrho_0 \sim \varrho_3$ are all positive constants that actually depend on the settings of the target bandit scenario. More details can be found in [25].

We do not "get Constraint (6) from Constraint (3)"; instead, in order to satisfy Constraint (3), we define a new constraint, (i.e., Constraint (6)), to replace Constraint (3), because we can prove that satisfying Constraint (6) will lead to the satisfaction of Constraint (3). We choose to construct Constraint (6) using the specific constants $\Xi_1(B)$ and $\Xi_2(B)$ for two reasons as

First, as stated above, our current form of Constraint (6) based on $\Xi_1(B)$ and $\Xi_2(B)$ can provably make Constraint (6) a sufficient condition for Constraint (3), which is illustrated in Theorem 1. Constraint (3) is $\mathcal{O}(\sum_{t\le\tau_B}\rho_t^{-\alpha}/\varepsilon) \le \tau_B$, and if we use ϖ_t to concisely represent the parameters in the big-$\mathcal{O}$, Constraint (3) can be rewritten as $\sum_{t\le\tau_B}\{\varpi_t\rho_t^{-\alpha}/\varepsilon - 1\} \le 0$. Because $\Xi_1(B) \le \tau_B \le \Xi_2(B)$, we can then define Constraint (6) as $\sum_{t\le\tau_B}\{\varpi_t\rho_t^{-\alpha}/\varepsilon - \Xi_1(B)/\Xi_2(B)\} \le 0$. Obviously, satisfying it makes us satisfy Constraint (3).

Second, one approach to satisfying the Constraint (3) is to ensure $\varpi_t\rho_t^{-\alpha}/\varepsilon - 1 \le 0$ for each one of the epoch $\forall t \le \tau_B$. However, the decision ρ_t is made at the beginning of each epoch t (i.e., before ϖ_t is revealed). That is, after firstly determining ρ_t and then seeing ϖ_t, it could turn out that the instantaneous constraint $\varpi_t\rho_t^{-\alpha}/\varepsilon - 1 \le 0$ could be then violated. To ensure $\varpi_t\rho_t^{-\alpha}/\varepsilon - 1 \le 0$ "as much as possible", we would like to adopt a more strict substitute (i.e., $\varpi_t\rho_t^{-\alpha}/\varepsilon - \Xi_1(B)/\Xi_2(B) \le 0$). Although $\varpi_t\rho_t^{-\alpha}/\varepsilon - \Xi_1(B)/\Xi_2(B) \le 0$ may still be violated, $1 - \Xi_1(B)/\Xi_2(B)$ could be regarded as a run-off gap to maintain $\varpi_t\rho_t^{-\alpha}/\varepsilon - 1 \le 0$ not violated. While there may exist other different approaches to satisfying Constraint (3), we have rigorously exhibited that our specific approach of using the current form of Constraint (6) with $\Xi_1(B)$ and $\Xi_2(B)$ to replace Constraint (3) works indeed.

Shown in the previous work [25], given budget B and bandit mechanism, the stopping time has a range (i.e., $\Xi_1(B) \le \tau_B \le \Xi_2(B)$), which guides us for decoupling the rest of $\mathbb{P}$ as

$$\begin{aligned} \min \quad & \sum_{t\le\tau_B}\sum_e x_{et}\{\sum_i \beta_{iet} + c_{et}\} \qquad [\mathbb{P}_2] \\ s.t. \quad & \sum_e x_{et} = 1,\ \forall t \le \tau_B, \quad \sum_{t\le\tau_B, e} r_{et}x_{et} \le B, \qquad (8) \\ var. \quad & x_{et} \in \{1,0\}, \end{aligned}$$

Algorithm 1 Structure of Online Schema

```
1: Initialize $t = 0$; Try all edges once, $\{p_e = 1, q_e = 0\}$;
2: Initialize $\boldsymbol{\lambda}_1 = \mathbf{0}$, and proper $\bar{\rho}_1, \gamma_1, \gamma_2$;
3: while $B \ge 0$ do
4:     $t = t + 1$;
5:     Call Algorithm 2;      //Control of Local Updates
6:     Call Algorithm 3;      //Control of Global Aggregations
7:     Update $B = B - r_{\bar{e}t}$; //Update Cost for Renting Edges
8: end while
9: $\tau_B = t$;                 //Stamp of Stopping Time
```

where all of these terms regarding x_{et} is decomposed from $\mathbb{P}$ under all other decisions fixed. For simplicity, if $\forall e$ is selected,

$$\psi_{et} \triangleq \sum_i \beta_{iet} + c_{et}, \quad \forall t \le \tau_B. \qquad (9)$$

Although the original problem is split into two parts, the blindness of the inputs on the fly hampers us from efficient solution, especially when the decisions of $\mathbb{P}_2$ are integers.

The decomposition is performed the following way. Before decomposition, $\mathbb{P}$ contains Constraints (1), (2), (3), and (4). After the decomposition, both the objective function of $\mathbb{P}$ and the domains Constraint (4) are naturally split into $\mathbb{P}_1$ and $\mathbb{P}_2$. Further, $\mathbb{P}_1$ contains Constraints (5) and (6), where Constraint (5) is from Constraint (1) of $\mathbb{P}$ and Constraint (6) is from Constraint (3) of $\mathbb{P}$; $\mathbb{P}_2$ contains Constraint (8) which is from Constraint (2) of $\mathbb{P}$. This process is shown in detail in Fig. 6.

We have also made changes to some constraints during this decomposition. In order to compose Constraint (5), we have relaxed Constraint (1) to a "long-term" format by applying the summation over time until τ_B to both sides of the inequality. To compose Constraint (6), we have introduced $\Xi_1(B)$ and $\Xi_2(B)$ into Constraint (3), where we observe $\Xi_1(B) \le \tau_B \le \Xi_2(B)$. We highlight that both the lower bound $\Xi_1(B)$ and the upper bound $\Xi_2(B)$ depend on B Such transformations from Constraints (1) and (3) to Constraints (5) and (6) all serve the purpose of our algorithm and theoretical analysis.

Our original purpose is to ensure all of the constraints $C1 \sim C3$, where Constraint (3) (i.e., $C3 : \tau_B \ge \mathcal{O}(\sum_{t\le\tau_B}\rho_t^{-\alpha}/\varepsilon)$) is derived from Lemma 1. Lemma 1 implies that as long as $C3$ is ensured, the global model convergence is then ensured. However, $\mathbb{P}$ is a mixed integer program and is hard to be tackled efficiently to obtain the optimum. Then, we decouple the original problem $\mathbb{P}$ into two subproblems $\mathbb{P}_1$ and $\mathbb{P}_2$, as shown in Fig. 6. Such decoupling is intuitive and it ensures that both of these two subproblems contain only one decision variable, respectively, and are relatively easy to be tackled.

B. Control of Local Model Updates

Within each epoch, Algorithm 2 first decides the number of local iterations trained on streaming data and then conduct the local model updates. We show these two parts as follows:

Iteration Numbers: When considering the problem of $\mathbb{P}_1$, these uncertain values $\{n_{it}\}$ hamper us from the optimum solution. Note that the number of local training iterations needs to be decided at the beginning of each epoch, which is used to control the local model updates on streaming data, and the changes of n_{it} may violate Constraint (5). If all of the inputs are given beforehand, $\mathbb{P}_1$ is considered to be solved as follows:

$$\min_{\tilde{\rho}_t \in \mathbb{R}^+} \max_{\lambda_t} \sum_{t \leq \tau_B} \big(f_t(\tilde{\rho}_t) + \boldsymbol{\lambda}_t^\top \boldsymbol{g}_t(\tilde{\rho}_t)\big), \quad (10)$$

where the form is derived from the equivalent convex-concave problem by introducing the Lagrange multiplier $\boldsymbol{\lambda}_t \succeq \mathbf{0}$.

In $\min_{\tilde{\rho}_t \in \mathbb{R}^+} \max_{\lambda_t} \sum_{t \leq \tau_B} \big(f_t(\tilde{\rho}_t) + \boldsymbol{\lambda}_t^\top \boldsymbol{g}_t(\tilde{\rho}_t)\big)$, f_t is our concise representation of the objective function of $\mathbb{P}_1$ and $\boldsymbol{g}_t$ is our concise and aggregated representation of the constraints. Therefore, every symbol, except the decision variables $\tilde{\rho}_t$ and $\boldsymbol{\lambda_t}$, represents the input. We use the concise representations for the ease of presentation. The issue here might be "when" we can have access to such inputs. The min-max formulation and concise representation here is for our original problem formulation $\mathbb{P}_1$. In the online setting, we make control decisions on the fly as the inputs reveal themselves gradually. For each epoch t when making decisions, only f_t and $\boldsymbol{g}_t$ can be observed and anything beyond the epoch t cannot be observed.

Note that both $f_t(\cdot)$ and $\boldsymbol{g}_t(\cdot) = [g_{1t}, g_{2t}]^\top$ here are convex functions with respect to $\tilde{\rho}_t \in \mathbb{R}^+$. To solve it in an online manner, the gradient incurred from the previous epoch can be used as a guidance intuitively. Thus, we can alternate between minimizing the objective with respect to the primal decision $\tilde{\rho}_{t+1}$ via a modified descent step and maximizing the objective with respect to the lagrange multiplier via a dual ascent step. More specifically, the modified primal step is conducted as

$$\min_{\tilde{\rho}_{t+1} \in \mathbb{R}^+} \ \mathcal{P}_{1,t} \qquad [\mathbb{P}_{1,t}]$$
$$\mathcal{P}_{1,t} = \nabla f_t(\tilde{\rho}_t)(\tilde{\rho}_{t+1} - \tilde{\rho}_t) + \boldsymbol{\lambda}_{t+1}^\top \boldsymbol{g}_t(\tilde{\rho}_{t+1}) + \frac{||\tilde{\rho}_{t+1} - \tilde{\rho}_t||^2}{\gamma_1},$$

while the Lagrange multiplier $\boldsymbol{\lambda}_{t+1}$ is updated as follows:

$$\boldsymbol{\lambda}_{t+1} = [\boldsymbol{\lambda}_t + \gamma_2 \boldsymbol{g}_t(\tilde{\rho}_t)]^+, \quad (11)$$

where γ_1 and γ_2 are both the step sizes of the online learning schema, and $[\cdot]^+$ refers to the term $\max\{\cdot, 0\}$ for each dimension. Actually, the gradient based approach mentioned before in the primal step is the approximation of f_{t+1} by linking two consecutive epochs. In each epoch, the prediction regarding ρ for next epoch is illustrated in Line 15 of Algorithm 2.

Since the results solved from $\mathbb{P}_{1,t}$ are reals, we need extra round step to obtain integer decisions, as shown in Line 16. More specifically, any round strategy could be applied as long as $E[\bar{\rho}_{t+1}] = \tilde{\rho}_{t+1}$. For example, $\bar{\rho}_{t+1}$ equals $\lfloor \tilde{\rho}_{t+1} \rfloor$ with the probability of $\lceil \tilde{\rho}_{t+1} \rceil - \tilde{\rho}_{t+1}$, otherwise equals $\lceil \tilde{\rho}_{t+1} \rceil$.

Training on Streaming Data: The parameters initialized are illustrated in Line 2 of Algorithm 1. The choice of proper step sizes are discussed in theoretical analysis later. Note that the initial value of $\bar{\rho}_1$ could be any feasible one. Based on the prediction from previous epoch and the parameters initialized, the training on streaming data for each device is illustrated in Lines 1 to 14 of Algorithm 2. We should mention here that Algorithm 2 needs to be called for each device along with the arrivals of the data within each epoch, and the pseudo code only shows the behaviour of $\forall$ device i and $\forall$ epoch $t \leq t_B$.

For those data that are not selected by sampling, such data do not contribute to related gradient calculation in Algorithm 2. The strategy of not using every single data sample for the gradient calculation is "sufficient" in the sense that we can already use this approach to design our algorithms and prove the corresponding theoretical performance guarantees. Using every single data sample may lead to a quite different design of related algorithms, which is out of the scope of our paper.

Algorithm 2 Local Model Updates, $\forall$ Device i, $\forall$ Epoch t

```
// Training on Streaming Data given $\bar{\rho}_t$
1:  Download latest global model $\boldsymbol{w}_0$, $\boldsymbol{\mathcal{S}} = \emptyset$, Buf $\leftarrow \emptyset$;
2:  for $j$ from 1 to $|\boldsymbol{\mathcal{M}_t}|$ do
3:      $\tilde{\boldsymbol{w}}_0 \leftarrow \boldsymbol{w}_{j-1}$;  Buf $\leftarrow$ Arrived $v_{ijt}$ data samples;
4:      for $z$ from 1 to $\bar{\rho}_t$ do
5:          if (Buf is non-empty) AND ($z$ is even) then
6:              Move a data sample $\aleph'$ from Buf to $\boldsymbol{\mathcal{S}}$; $\boldsymbol{\delta}(\aleph') = \mathbf{0}$;
7:              $\boldsymbol{A} \leftarrow \sum_{s \in \boldsymbol{\mathcal{S}}} \boldsymbol{\delta}(s)/|\boldsymbol{\mathcal{S}}|$;
8:          end if
9:          Sample one piece of data $\aleph$ uniformly from $\boldsymbol{\mathcal{S}}$;
10:         Compute gradient $\boldsymbol{\sigma}$ upon $\mathcal{L}$, $\tilde{\boldsymbol{w}}_{z-1}$ and $\aleph$;
11:         $\tilde{\boldsymbol{w}}_z \leftarrow \tilde{\boldsymbol{w}}_{z-1} - \eta(\boldsymbol{\sigma} - \boldsymbol{\delta}(\aleph) + \boldsymbol{A})$;
            $\boldsymbol{\delta}(\aleph) \leftarrow \boldsymbol{\sigma}$;
12:     end for
13:     $\boldsymbol{w}_j \leftarrow \tilde{\boldsymbol{w}}_{\bar{\rho}_t}$;
14: end for
// Update $\bar{\rho}_{t+1}$
15: $\boldsymbol{\lambda}_{t+1}, \tilde{\rho}_{t+1}$ are updated by (11) and $\mathbb{P}_{1,t}$, respectively;
16: $\bar{\rho}_{t+1} \leftarrow Round(\tilde{\rho}_{t+1})$;
```

At any time stamp j in epoch t, the mini-batch of v_{ijt} data samples arrive at the buffer of the device i, as in Line 3 of Algorithm 2, and we use such data to do the random sampling, gradient calculation, and model update. And, at the next time stamp $j+1$, the mini-batch of v_{ij+1t} data samples will arrive at the buffer of the device i and will override the entire buffer which held the mini-batch of the v_{ijt} data samples just now.

For all of the data arrived, Algorithm 2 only chooses one data sample to calculate the gradient per local iteration, as shown in Line 10, where the gradient $\boldsymbol{\sigma}$ is calculated based on the loss function $\mathcal{L}$, data sample $\aleph$ and currently maintained model $\tilde{\boldsymbol{w}}$. After that, the gradient is used to update $\tilde{\boldsymbol{w}}$, as shown in Line 11 of Algorithm 2. Such update uses both of the data sampled (i.e., the gradient $\boldsymbol{\sigma}$ just calculated), and all data maintained by $\mathcal{S}$ (i.e., the averaged weight vector $\boldsymbol{A}$).

Line 11 of Algorithm 2 has two "assignment" equations. In the first equation (i.e., the model update equation), $\boldsymbol{\sigma}$ is the "current" gradient which is calculated in Line 10 with the newly-sampled data sample from Line 9, and $\boldsymbol{\delta}(\aleph)$ records the "previous" gradient which was calculated before the newly-sampled data sample (note that $\boldsymbol{\delta}(\aleph)$ can also be zero if the newly-sampled data sample happens to be the same one that has just been moved into the set $\boldsymbol{\mathcal{S}}$ as in Line 6). The second equation in Line 11 of Algorithm 2 records the current gradient $\boldsymbol{\sigma}$ in $\boldsymbol{\delta}(\aleph)$, so that $\boldsymbol{\delta}(\aleph)$ can be used in future iterations of the loop. Therefore, in the model update equation, there is no guarantee that $\boldsymbol{\sigma}$ and $\boldsymbol{\delta}(\aleph)$ are always equal.

z is an index, or a counter (i.e., integer), from 1 to $\bar{\rho}_t$. Thus, z is not always even; it becomes odd and even alternately, and only when it is even (i.e., Line 5 of Algorithm 2), we execute Line 6 of Algorithm 2. We use the self-incremental counter z

Algorithm 3 Global Model Aggregations

```
1: Find edge $\bar{e}$ with maximal $-\mu(\psi_{e1},...,\psi_{et-1})+q_e$;
2: $p_{\bar{e}} \leftarrow p_{\bar{e}}+1$;
3: $q_{\bar{e}} \leftarrow \{\sqrt{2\log t/p_{\bar{e}}}(1+1/r_{min})\}/\{r_{min}-\sqrt{2\log t/p_{\bar{e}}}\}$;
4: Choose edge $\bar{e}$ for global model aggregation;
5: Transfer the latest model to each device;
6: Return $r_{\bar{e}t}$ to **Algorithm 1**;
```

to control the move of the arriving data to the set $\boldsymbol{\mathcal{S}}$. Because we only move a data sample when z is even, we at most move $\bar{\rho}_t/2$ data samples to the set $\boldsymbol{\mathcal{S}}$. One can change the condition in Line 5 of Algorithm 2 in order to change the number of data samples moved, which will directly impact the local model convergence (i.e., the constant "2" as in Inequality (0d)).

Remarks: We should mention here that all of the weight vectors maintained only for those data samples arrived and kept in the set $\boldsymbol{\mathcal{S}}$. And the data samples arrived are chosen from the buffer Buf to $\boldsymbol{\mathcal{S}}$. Such process actually decreases the volume of data updated and maintained simultaneously. The whole training part for streaming data on a device is derived from strSAGA, which ensures the local model convergence after $\bar{\rho}_t$ local iterations, shown in Inequality (0d).

C. Control of Global Model Aggregation

This part involves the collaboration between Algorithm 1 and Algorithm 3. Algorithm 3 conducts the bandit plays per epoch for the location regarding the global model aggregation, considering both of exploration and exploitation. Algorithm 1 then updates the left budget to control the stopping time.

Requirement on Constrained Budget: The value in terms of the constrained budget needs to be large enough to support sufficient global model aggregations while the value of budget should be as small as possible to decrease the overall cost.

Lemma 2. *Given budget B, the relationship between stoping time τ_B, its lower bound $\Xi_1(B)$ and upper bound $\Xi_2(B)$ is*

$$\sum_{t\le\tau_B}\frac{\Xi_1(B)}{\Xi_2(B)}=\tau_B\frac{\Xi_1(B)}{\Xi_2(B)}\le\tau_B=\sum_{t\le\tau_B}1,$$

where $\Xi_1(B)/\Xi_2(B)$ is the substitute we use to construct Constraint (6), as the sufficient condition of Constraint (3).

Proof. See Appendix B, using the definition of Ξ_1 and Ξ_2. □

Replacing Constraint (3) by its sufficient condition: Constraint (6), one of the results in our theoretical analysis (based on Lemma 2, shown later) is that, from the perspective of the expectation, the violation of Constraint (3) is ensured (i.e., the time-average violation will vanish as time goes to infinity).

UCB based Plays: If we know all of the inputs beforehand, we can choose the best edge for conducting all of the global aggregations. However, the variables are stochastic, the dynamic changes hamper us from precisely estimating the confidential interval of the inputs. Then, we propose to use the upper confidence bound as a substitute to estimate such inputs (i.e., the following inequality holds with high probability):

$$\mu(\psi_{e1},...,\psi_{et-1})-\mu(\psi_{e1},...,\psi_{e\tau_B})\le q_e, \tag{12}$$

where $\mu(\cdot)$ is the average function to obtain the corresponding average value. Here, $\forall e,\forall t\le\tau_B:\mu(\psi_{e1},...,\psi_{et})$ is

$$\mu(\psi_{e1},...,\psi_{et})=\frac{1}{|\mathcal{T}_{et}|}\sum_{j\in\mathcal{T}_{et}}\psi_{ej}, \tag{13}$$

where $\mathcal{T}_{et}$ denotes the set of epochs less than or equal to t that select edge e to conduct the global model aggregation. $|\cdot|$ is the number of the elements in $\mathcal{T}_{et}$ (i.e., selected epochs).

Actually, the average value of $\mu(\psi_{e1},...,\psi_{e\tau_B})$ is unavailable beforehand. Then, we use $\mu(\psi_{e1},...,\psi_{et-1})-q_e$ as the guidance, where q_e here is the bias. $q_e=\{\sqrt{2\log t/p_e}(1+1/r_{min})\}/\{r_{min}-\sqrt{2\log t/p_e}\}$, where t is the current epoch; p_e is a counter to indicate the number of times that the edge e has been selected so far; and r_{min} is a lower bound of the minimal renting cost over edges, assumed known in prior.

Considering the objective is to obtain the minimum, we have the following inequality that holds with high probability:

$$-\mu(\psi_{e1},...,\psi_{e\tau_B})\le-\mu(\psi_{e1},...,\psi_{et-1})+q_e, \tag{14}$$

where $-\mu(\psi_{e1},...,\psi_{et-1})+q_e$ actually leads to an upper bound of $-\mu(\psi_{e1},...,\psi_{e\tau_B})$ with high probability. Such inequality guides Algorithm 3 to choose the most suitable edge for the global aggregation in epoch t, as shown in Line 1 of Algorithm 3 (i.e., the edge with the maximal upper bound):

$$\bar{e}=argmax_e\{-\mu(\psi_{e1},...,\psi_{et-1})+q_e\},\ i.e., \bar{x}_{\bar{e}t}=1. \tag{15}$$

The update of the bias q_e relies on the trade-off between exploration and exploitation, where it takes the number of being selected into consideration. The number of edge selections is recorded by using p_e, which is updated shown in Line 2 of Algorithm 3. With the increase of the selection number p_e, Algorithm 3 has more confidence to choose it, since it has tried multiple times and obtained adequate information. Thus, with the increase of the selection number p_e, the bias decreases.

We consider the devices, the edges, and the system administrator (which could be the cloud or one fixed edge) in our system. After completing the last time stamp in the epoch t, each device notifies the system administrator in order to start the global model aggregation. Then, the system administrator decides a specific edge $\bar{e}$, and sends this selection result back to all the devices for the global model aggregation for t. Afterwards, every device i sends $a_{i\bar{e}t}$ and $b_{i\bar{e}t}$, and the edge $\bar{e}$ sends $c_{\bar{e}t}$ to the administrator. That is, after conducting the global aggregation for t, the system administrator will have received $\{a_{ie1},a_{ie2},...,a_{iet}\}$, $\{b_{ie1},b_{ie2},...,b_{iet}\}$, $\{c_{e1},c_{e2},...,c_{et}\}$ for all e that have been selected for at least once up until t. Based on this, the system administrator itself maintains the corresponding p_e and q_e, which will be used to select the edge at the next epoch $t+1$. We highlight that these communications themselves are fixed control flows, which we do not optimize.

IV. Theoretical Analysis

We first study the theoretical results by using online learning and bandit play, respectively. After that, we combine them together in order to obtain the main theorems, which show the guarantee of the global model convergence and the regret compared with the optimum. And the assumptions used are

Assumption 1: All loss functions $\{\mathcal{L}\}$ are convex.
Assumption 2: The domain is bounded by value G_d and the gradients are bounded by $\nabla f_t \leq G_f$ and $||\nabla \boldsymbol{g}_t|| \leq G_g, \forall t$.
Assumption 3: There exists a constant $\varsigma > 0$, and an interior point $\hat{\rho}_t$ such that $\boldsymbol{g}_t(\hat{\rho}_t) \preceq -\varsigma \mathbf{1}$ and $\varsigma > \hat{V}(\boldsymbol{g})$, where

$$\hat{V}(\boldsymbol{g}) \triangleq \max_t \max_{\tilde{\rho}_t} ||[\boldsymbol{g}_{t+1}(\tilde{\rho}_t) - \boldsymbol{g}_t(\tilde{\rho}_t)]^+||.$$

Assumption 1 and 2 are very common and are widely used. Assumption 3 actually ensures the existence of the optimum and the feasible region defined by $\boldsymbol{g}_t(\tilde{\rho}_t) \preceq \mathbf{0}$ is large enough or the trajectory is smooth enough across time.

Lemma 3. *By using the prediction on $\{\bar{\rho}_t\}$ over time upon online learning, the following regret and fit hold:*

$$reg_o = E[\textstyle\sum_{t\leq\tau_B} f_t(\bar{\rho}_t)] - \sum_{t\leq\tau_B} f_t^* \leq \Omega_1,$$
$$fit_o = ||[E[\textstyle\sum_{t\leq\tau_B} \boldsymbol{g}_t(\bar{\rho}_t)]]^+|| \leq \Omega_2,$$

where Ω_1 and Ω_2 are sub-linear growth with respect to B.

Proof. See Appendix C, with the help of Lemma 2. □

"Fit" measures the violation of the constraints. If the fit is positive, then the constraints are violated (the larger the fit is, the more the constraints are violated). If the fit is less than or equal to zero, then the constraints are respected. Our theoretical analysis in this paper shows that the fit grows only sub-linearly as time goes, which means the increase of the fit is much slower than the progress of time; in other words, the time-average fit will become zero (i.e., the violation will vanish in time-average sense) if time goes to positive infinity.

Lemma 4. *By using the bandit plays in terms of $\{\bar{x}_{et}\}$ over time, the following regret holds (ψ_{et} contains a_{iet}, b_{iet}, c_{et}):*

$$reg_b = E_{\xi\sim\mathscr{D}}[\sum_{t\leq\tau_B}\sum_{e}\bar{x}_{et}\psi_{et} - \sum_{t\leq\tau_B}\psi_{e^*t}] \leq \Omega_3,$$

where e^* is the optimum edge and Ω_3 is sub-linear growth.

Proof. See Appendix D, with the help of Lemma 2. □

In Lemma 3: Ω_1 equals $\Xi_2(B)^{\nu_1}$ where $\nu_1 < 1$. Via Equation (7), $\Omega_1 = (2\varrho_0 B + \varrho_1)^{\nu_1}$. In Lemma 3: $\Omega_2 = \Xi_2(B)^{\nu_2} + \Gamma$, where $\nu_2 < 1$; Γ is a constant. Also, $\Omega_2 = (2\varrho_0 B + \varrho_1)^{\nu_2} + \Gamma$. In Lemma 4: $\Omega_3 = \varrho_0 B - \Xi_1(B) = \varrho_2 + \varrho_3 log(\varrho_1 + \varrho_0 B)$. Here, $\Omega_1 \sim \Omega_3$ are all sub-linear growth with respect to B.

Theorem 1. *The global model convergence is ensured as*

$$E[\frac{\sum_{t\leq\tau_B} l_t}{\tau_B} - l_{opt}] \leq \varepsilon.$$

Proof. See Appendix E, using Lemma 1 to Lemma 3. □

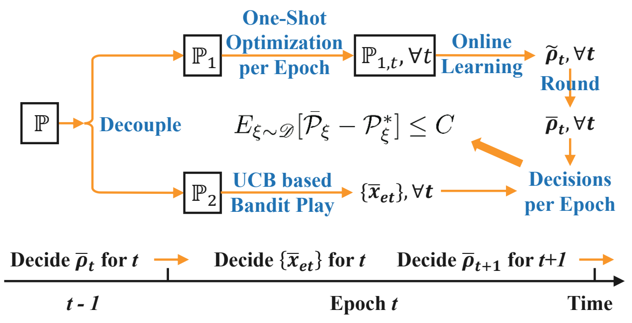
Theorem 2. *The regret in terms of the objective value obtained by using our online schema and the optimum one is*

$$regret = E_{\xi\sim\mathscr{D}}[\bar{\mathcal{P}}_\xi - \mathcal{P}_\xi^*] \leq \mathcal{O}(\tau_B^{\nu_1} + log\tau_B),$$

where ν_1 is less than 1 as shown later and the two terms on the right are both sub-linear respect to stopping time τ_B.

Proof. See Appendix F, using Lemma 2 to Lemma 4. □

To derive the regret, we do not need to access the optimum solution; rather, we use the lower and the upper bounds of the optimum as the substitutes to facilitate and complete the proof. We can often derive such lower and upper bounds even if we cannot access the optimum solution itself, which is a pretty common practice in lots of online learning and bandit learning research. If we were to be able to access the optimum, then there would be no need to derive the regret because we would be already "optimum". We elaborate some details below.



$\mathbb{P}$
Decouple
$\mathbb{P}_1$
One-Shot Optimization per Epoch
$\mathbb{P}_{1,t}, \forall t$
Online Learning
$\tilde{\rho}_t, \forall t$
Round
$\bar{\rho}_t, \forall t$
$E_{\xi\sim\mathscr{D}}[\bar{\mathcal{P}}_\xi - \mathcal{P}_\xi^*] \leq C$
$\mathbb{P}_2$
UCB based Bandit Play
$\{\bar{x}_{et}\}, \forall t$
Decisions per Epoch
Decide $\bar{\rho}_t$ for t
Decide $\{\bar{x}_{et}\}$ for t
Decide $\bar{\rho}_{t+1}$ for $t+1$
t - 1
Epoch t
Time

Fig. 7: RoadMap for Theoretical Analysis

(i) For local model updates for each epoch, the optimum ρ_t^* under integers is substituted by the optimum $\widetilde{\rho}_t^*$ under reals, as shown in Lemma 3. Note that the real domain contains the integral domain, and for a minimization problem, the optimum objective in a larger domain is actually no greater than the optimum objective obtained in a smaller domain.

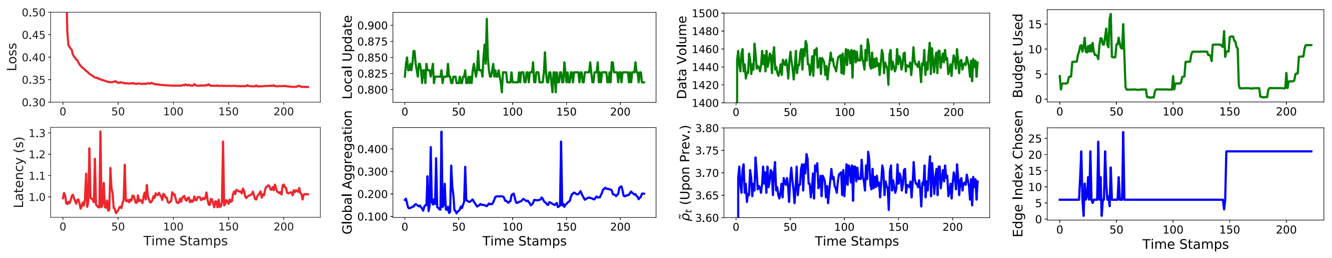
(ii) For global model aggregations at edge, the regret regarding the cumulative renting costs and its optimum is obtained by the Upper Confidence Bound (UCB) based analysis, similar to a typical Multi-Armed Bandit (MAB) setting. Essentially, in this analysis, the gap between a selected non-optimum edge and the optimum edge per epoch is upper bounded.

(iii) Combining the previous two regrets together, we complete the proof of Theorem 2, which implies the overall regret is bounded. This proof is further completed by introducing multiple complementary terms, whose sum is 0. These complementary terms split the target regret into multiple sub-regrets that are exactly the ones used in the local model updates and the global model aggregations, respectively.

V. Experimental Evaluations

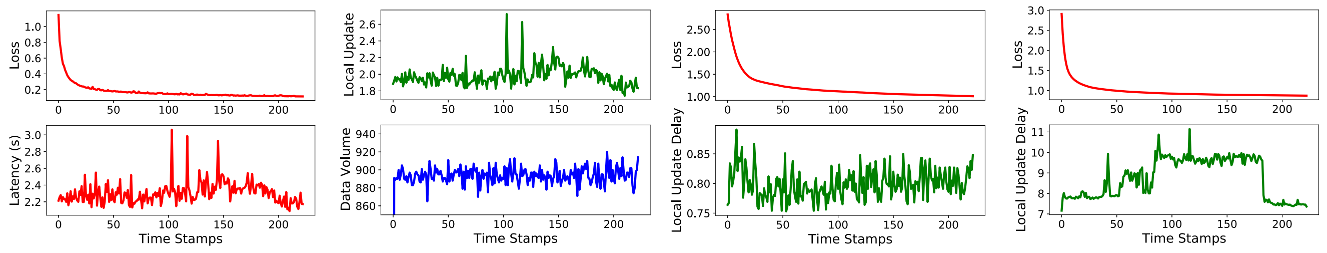
A. Data and Settings

Streaming Data: We use the four commonly used datasets: a9a [17], rcv [18], movielens100k [19], and movielens1m [19]. More specifically, a9a refers to a binary classification of the UCI adult dataset, which actually contains 32561 samples and 123 features; rcv refers to the binary classification of Reuters articles by topics, which contains 20242 samples and 47236 features; movielens100k refers to the matrix factorization of 100K movie ratings from 943 users on 1682 movies (i.e., 1682 features); movielens1m refers to the matrix factorization of 1M movie ratings from 6040 users on 3952 movies. We convert these training data into the streams, which obey the Poisson distribution. To mimic the realistic generation of data samples, these datasets are also shuffled randomly for different epochs. The training data arrive over 50 time steps within each epoch, where an epoch lasts at least 20 seconds or more.



(a) Loss and Latency (b) Latency Details (c) Update of $\bar{\rho}_t$ (d) Bandit Plays

Fig. 8: Runtime Details of Our Proposed Online Approach (a9a)



(a) Loss and Latency (rcv) (b) Summary of Details (rcv) (c) movielens100k (d) movielens1m

Fig. 9: Runtime Details of Our Proposed Online Approach (Other Datasets)

Local Training: We adopt two widely-used loss functions for model training: logistic regression (convex) and matrix factorization (non-convex). The number of the participant mobile devices varies as 2∼32. The parameters used for streaming data learning are derived as $\alpha = 0.75$ and $\mathcal{H} = 0.001$ [11].

Global Aggregations: We set the locations and the time-varying bandwidths of 28 edges [20] by tracing file downloads between the AliCloud and those nearby edges, ranging from 155KB/s to 3939KB/s. The Round-Trip Times (RTTs) between the users and those edges are set as 81∼1413 ms [21], while the average costs for renting edges are derived from Amazon [3], ranging from 10.45%∼1320% of the normal price. The corresponding dynamic changes over time are derived from Microsoft [26], ranging from 0.025∼0.999.

We set the loss gap of federated learning as 0.005 [11]. During the global model aggregations, FedAvg is used [22], where the parameters of the model being trained are aggregated and averaged. All the inputs, including the bandwidths, the RTTs, and the aggregation executions, are fed to the problem on the fly and remain unknown before revealed. Note that all of the variations over time are directly derived from real-world traces.

Implementation: We have implemented all the algorithms in about 4000 lines of Python codes on top of an existing implementation [11]. The experiments are conducted upon a Dell desktop, a Dell PowerEdge R740 server, and an Inspur SN5160M4 server. The latter two are used for the large-scale experiments. The optimization subproblems in our proposed approach are solved using standard tools of AMPL and IPOPT.

Algorithms for Comparison: We consider the loss and the latency for training as the two primary performance metrics, and compare multiple combinations of different algorithms.

First, for local model updates, we consider the following:

- *BGD* calculates the gradient for model updates per time by thoroughly processing the entire dataset.
- *SGD* calculates the gradient for model updates per time by only processing a single data sample.
- *mini-batch SGD* calculates the gradient for model updates per time by processing a subset (i.e., a batch) of the whole dataset. Within each local round, mini-batch SGD makes one pass of all of the batches. The number of local rounds used for batch training approaches ranges from 1∼30 [1].

Second, for deciding the number of local iterations during local model updates (i.e., ρ_t), we consider two approaches:

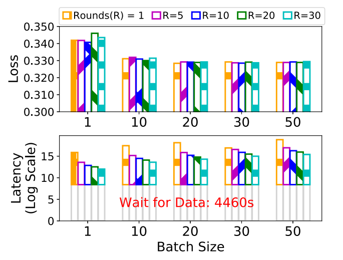
- *Fixd value* uses multiple ρ_t candidates, which are fixed at the very beginning and are used during the training of local model updates, ranging in 1∼20.
- *strSAGA* [11] exactly uses the settings used in previous state-of-the-art work, where the number of local iterations for local model updates is hundreds or even more.

Third, for selecting edges for global aggregations, we have

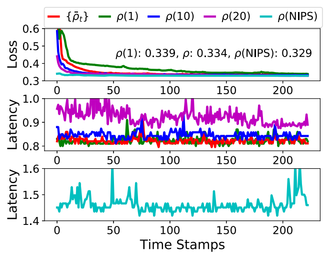
- *minTrans* chooses the edge that has the minimal transmission cost in terms of the posterior average.
- *minCost* chooses the edge that has the minimum renting cost in terms of the posterior average.
- *ran* chooses the edge randomly from all the candidates.
- *E* chooses an edge for all epochs from the edge set that has already been determined by our proposed approach.

B. Evaluation Results

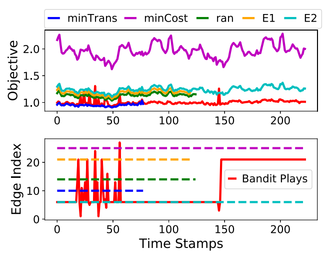
Effectiveness of Our Approach: Fig. 8 shows the loss and the latency of our approach upon the a9a dataset. During the arrivals of the streaming data, our approach through using local model updates with global model aggregations reaches the convergence of the global model, as in Fig. 8(a). Furthermore, the cumulative latency incurred in total is about 200 seconds, which is acceptable and is amortized over the time. The peak latency incurred spends at most 1.3 seconds. Fig. 8(b) further shows the detailed latency for local model updates and global model aggregations, respectively. During the execution of local



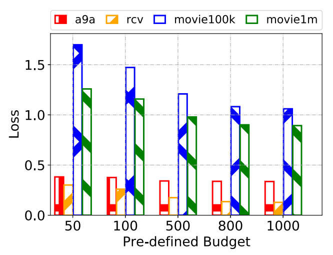
(a) Batch Training Algorithms



(b) Algorithms for $\{\rho_t\}$

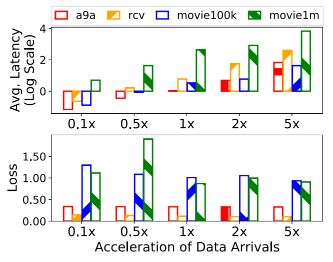


(c) Algorithms for Aggregations

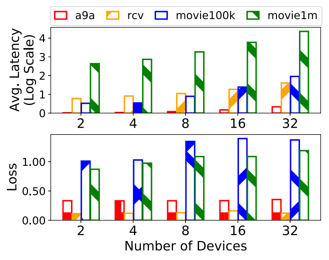


(d) Impact of Budget

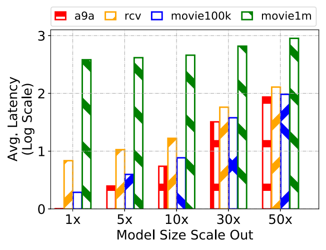
Fig. 10: Empirical Superiority over Other Algorithms



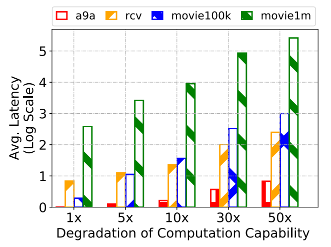
(a) Impact of Arrivals



(b) Impact of Devices



(c) Impact of Model Sizes



(d) Impact of Computation

Fig. 11: Impact of Various Configurations

model updates over time, the number of local iterations is dynamically adjusted to capture the changes on the volume of streaming data. Note that the real values of $\tilde{\rho}_t$ calculated in Fig. 8(c) use the information of the data volume revealed from the previous epoch (i.e., $\mathbb{P}_{1,t}$). During the global model aggregations, our approach also chooses the most suitable edge based on maintained upper confidence bound, as in Fig. 8(d).

Fig. 9 shows the performance of our approach when applied to other datasets. It confirms the global model convergence, even for one million data samples. The latencies incurred for all of these datasets are acceptable (i.e., only several seconds within each epoch). Although the computation is proportional to the data volume in each epoch, it only involves the sum of average and the calculation of the gradient on each single data sample. Thus, the peak load is light, and the cumulative latency is only several minutes, much less than the whole time horizon (i.e., 74 minutes in total (at least 20 seconds per epoch)).

Advantages of Our Approach over Alternatives: Fig. 10 exhibits the empirical superiority of our approach over other state-of-the-art approaches. Fig. 10(a) shows the loss and the latency incurred by these batch training approaches. Note that the latency has already included the delay of calculating the loss over test data samples after each model update. BGD and SGD can be summarized into the scope of mini-batch SGD when the batch size is set as the size of the whole dataset and as 1, respectively. Note that the numbers in the labels of the legend refer to the number of rounds, and each round processes all of the batches. Those batch training approaches have to wait for the preparation of the entire dataset and hence waste time. These approaches also spend much time on local training. To reduce the loss to the desired level, hundreds of seconds, or even tens of hundreds of seconds are needed.

Fig. 10(b) visualizes the comparison between the multiple choices of the local iteration ρ_t. The red curve indicates our approach, and ρ (NIPS) refers to the one from strSAGA. If the number of local iterations is small (i.e., 1 iteration), the loss of 0.339 is achieved. For our approach, by using at most 1 second on the local model updates per epoch, the loss improvement is desirable. With the increase of ρ_t, in order to achieve the same improvement on the loss, the latency involved is doubled compared to our proposed approach. Even for the ρ_t designed by strSAGA, whose value is hundreds or more, the maximal improvement on the loss is limited (i.e., at most 0.005).

Fig. 10(c) illustrates the comparison between multiple approaches regarding the selection of edges for the global model aggregations. The red curve shows the results of our bandit plays, and E1 and E2 choose two edges during the whole training lifecycle from the set that are determined by our bandit plays. When those edges are selected based on the minimal transmission cost or the minimal renting cost, the number of global aggregations could be insufficient for the global model to converge. When the edges are selected from the decisions made by our bandit plays, if the decisions are not switched to adapt to the variations of transmission cost and renting cost, the overall latency incurred is much higher than that of ours.

Fig. 10(d) further shows various choices of the constrained budget. With the increase of such budget, the global loss decreases dramatically. However, when the budget reaches to 1000 or more, the improvement becomes slight. According to the lower bound of τ_B for sufficient global model convergence and sufficient bandit plays, the desired budget is about 1300 based on $argmin_B \Xi_1(B)$, which is suitable for both of the global model convergence and the resource saving.

Scalability of Our Approach: Fig. 11 illustrates the scalability of our approach. Fig. 11(a) varies the data arrivals by adjusting the "acceleration" of the streams. With the growth of the acceleration, more data arrive within an epoch, resulting in more stable loss and higher latency for local model updates. When the acceleration is small, the data volume is insufficient for reaching the desired loss. Note that the latency upon the movielens1m data is high due to the tens of millions of data samples, and we use the log scale for illustration. Although the data volume is high, the overall latency is still controlled within one hundred seconds, which is acceptable.

Fig. 11(b) changes the number of those mobile devices participating in the training. With the growth of the participant number, the overall latency is increased. Note that we calculate the overall latency including both local model updates and global model aggregations. All of the data samples are randomly shuffled over these devices. As a result, with the growth of devices, data samples are distributed more unevenly over devices. Further, the loss of the global model being trained also increases a little due to the skewed data distribution.

Fig. 11(c) shows the influence regarding the transmission scalability. We also vary the model size to mimic the scenario where large models are used. With the growth of the model size, the overall latency in terms of transmission increases, but the overall delay within an epoch is also controlled within a hundred seconds on average. For the scenario where an epoch is of tens of minutes or more, the overall latency for the whole training process is also lightweight for the devices.

Fig. 11(d) shows the influence regarding the computation scalability. We vary the computation capability for both the mobile devices and the edges. Here, the computation capability is considered as the speed of processing data samples within one unit time, and the scalability refers to the degradation of computation capability. When the degradation is high, the overall latency increases for both local model updates and global model aggregations. However, the overall latency is still suitable since local updates only calculate the gradient upon one data sample per iteration and other computations only incur weighted sums on both mobile devices and edges.

VI. Related Works

We summarize prior research in two categories, and highlight their drawbacks compared to our work, respectively.

Streaming Learning: Some works focused on the incremental methods regarding the gradient updates [27]. For different kinds of loss functions, SAG [28], SDCA [7], SVRG [8], etc., were proposed to achieve fast linear convergence rates in an incremental way. SAGA [9] supported non-strongly convex problems and was adaptive to inherent strong convexity of the problem. ETH [10] proposed an algorithm for dynamically increasing the effective sample size under the scenario, where making multiple passes through the full dataset was prohibitive. Upon SAGA, strSAGA [11] presented an approach for maintaining the model over samples that arrived over time.

Although those works have studied the learning over streaming data, distributed learning across multiple devices is not considered. They also do not jointly investigate the problem with federated learning, for which the model updates, model aggregations and model convergence need to be explored.

Federated Learning at Edges: Many works studied both of the optimization for local training and the efficiency for global aggregations. Google [29] analyzed multiple local minimizers for model convergence. Wang $et\ al.$ [22] controlled the frequency of global model aggregations given a resource budget of edges. Tu $et\ al.$ [12] designed the network-aware optimization of federated learning. Jin [13] and Luo [14] focused on cost-effective federated learning design, through controlling both participants and aggregations. Other works [15, 16] further proposed algorithms for federated learning over wireless or edge networks, improving the efficiency during the training.

Those works usually start the training process on the data only after the preparation of the entire dataset. Specifically, no algorithms of federated learning have been designed for training the model along with the dynamic arrivals of the data.

VII. Conclusion

In order to avoid the training load congestion after the preparation of the entire training data, model training should be amortized along with the dynamic arrivals of the streaming data. In this paper, we formulate the problem by considering both local model updates on streaming data and global model aggregations of federated learning over edge networks. We build a non-linear mixed integer program for minimizing the long-term cumulative latency while guaranteeing the maximal training load and global model convergence. We design an online approach to control these two components via online learning and bandit play techniques. We rigorously prove the regret regarding the objective and the global model convergence. Our trace-driven experiments confirm the advantages of our approach over other approaches in practice.

Appendix

In this section, we show all necessary proofs in detail.

A. Proof of Lemma 1

Proof. We study the relationship between the global optimum loss l_{opt} and the local optimum. Since the global optimum is revealed under the scenario where the model is given and fixed for all loss functions, we can replace it with those local optimum models to obtain a better loss. That is

$$l_{opt} \geq \frac{\sum_{i,t} l^*_{it}}{\tau_B N}.$$

Then, the upper bound of the left part in Lemma 1 is

$$E[\frac{\sum_t l_t}{\tau_B} - l_{opt}] \leq E[\frac{\sum_t l_t}{\tau_B} - \frac{\sum_{t,i} l^*_{it}}{\tau_B N}],$$

which means we only need to focus on the term $l_t - l^*_{it}/N$.

Since the loss function is convex and the aggregated model is the average sum of all local models trained on devices, then

$$l_t \leq \frac{\sum_i l^U_{it}}{N},\ \forall t \leq \tau_B.$$

Then, we only need to focus on the term $l^U_{it} - l^*_{it}$. Note that according to preliminary theoretical results [11], the following inequality holds by using local model updates (i.e., strSAGA):

$$E[l^U_{it} - l^*_{it}] \leq u \cdot \max\{1, (\frac{2n_{it}}{M_t \bar{\rho}_t})^{\alpha}\} \cdot \mathcal{H}_{it}(n_{it}) \triangleq \mathcal{A}_{it}\mathcal{H}_{it}(n_{it}),$$

where l^U_{it} is the loss incurred by the local model updates on device i in epoch t; l^*_{it} is the best loss achieved by the oracle; and $\mathcal{H}_{it}$ is the loss obtained by the empirical risk minimizer.

After combing previous inequalities together, we have

$$E[\frac{\sum_t l_t}{\tau_B} - l_{opt}] \leq E[\frac{\sum_{i,t}(l_{it}^U - l_{it}^*)}{N\tau_B}] \leq \frac{\mathcal{H}_{max}\sum_{i,t}\mathcal{A}_{it}}{N\tau_B} \leq \varepsilon,$$

where $\mathcal{H}_{max} = \max_{i,t}\{\mathcal{H}_{it}(n_{it})\}$. Via solving the minimum τ required in training to hold previous inequality, we have

$$\tau_B \geq \frac{\mathcal{H}_{max}\sum_{i,t}\mathcal{A}_{it}}{\varepsilon N} = \frac{u\mathcal{H}_{max}\sum_{i,t}\max\{1,(\frac{2n_{it}}{M_t\bar{\rho}_t})^{\alpha}\}}{\varepsilon N},$$

where the right part $\mathcal{O}(\sum_{t\leq\tau_B}\bar{\rho}_t^{-\alpha}/\varepsilon)$ is for ease of presentation. And the parameters ϖ_t are actually used. □

B. Proof of Lemma 2

Proof. Preliminary works [25] have revealed that given budget B, the following inequality holds regarding the stopping time:

$$\Xi_1(B) \leq \tau_B \leq \Xi_2(B),$$

where $\Xi_1(B) \triangleq \varrho_0 B - \varrho_2 - \varrho_3 log(\varrho_1 + \varrho_0 B)$, $\Xi_2(B) \triangleq 2\varrho_0 B + \varrho_1$, and ϱ_0 to ϱ_3 here are all constants. Thus, we have

$$\tau_B \geq \Xi_1(B) \geq \frac{\tau_B}{\Xi_2(B)}\Xi_1(B) \geq \frac{\sum_{t\leq\tau_B}\Xi_1(B)}{\Xi_2(B)}.$$ □

C. Proof of Lemma 3

Proof. We first study the fit fit_o, which measures the violation of the constraints. We have the following inequality:

$$||[E[\sum_{t\leq\tau_B}\boldsymbol{g}_t(\bar{\rho}_t)]]^+|| \leq ||E[\sum_{t\leq\tau_B}\boldsymbol{g}_t(\bar{\rho}_t)]||,$$

since adopting $[\cdot]^+$ for each dimension would only decrease the absolute value (e.g., a negative value is converted to be 0 after applying $[\cdot]^+$). Then we use the linear property of the expectation and have the following inequality:

$$fit_o \leq ||E[\sum_{t\leq\tau_B}\boldsymbol{g}_t(\bar{\rho}_t)]|| \leq ||\sum_{t\leq\tau_B}E[\boldsymbol{g}_t(\bar{\rho}_t)]||.$$

Note that $\boldsymbol{g}_t$ is convex for each of its dimension (i.e., $g_{1t}(\cdot)$ is linear with respect to its input under reals; $g_{2t}(\cdot)$ is convex with respect to its input under reals). We should mention here that the function $(\cdot)^{-\alpha}$ under reals is convex. We have

$$E[\boldsymbol{g}_t(\bar{\rho}_t)] \leq \boldsymbol{g}_t(E[\bar{\rho}_t]) + \Gamma,$$

where Γ is a constant, by adopting the Jensen Gap mentioned in [30]. Combining previous two inequalities together, we have

$$||\sum_{t\leq\tau_B}E[\boldsymbol{g}_t(\bar{\rho}_t)]|| \leq ||\sum_{t\leq\tau_B}\boldsymbol{g}_t(E[\bar{\rho}_t])|| + \Gamma,$$

where the inequality holds after adopting the Triangle Inequality over the 2-norm. Note that, the rounding part regarding $\tilde{\rho}_t$ in Algorithm 2 ensures $E[\bar{\rho}_t] = \tilde{\rho}_t$. We further have

$$fit_o \leq ||\sum_{t\leq\tau_B}\boldsymbol{g}_t(E[\bar{\rho}_t])|| + \Gamma \leq ||\sum_{t\leq\tau_B}\boldsymbol{g}_t(\tilde{\rho}_t)|| + \Gamma.$$

Since the regret reg_o is a linear function with respect to its input under reals, we have the following equation:

$$reg_o = E[\sum_{t\leq\tau_B}f_t(\bar{\rho}_t)] - \sum_{t\leq\tau_B}f_t^* = \sum_{t\leq\tau_B}f_t(E[\bar{\rho}_t]) - \sum_{t\leq\tau_B}f_t^*.$$

After applying $E[\bar{\rho}_t] = \tilde{\rho}_t$ again, we have

$$reg_o = \sum_{t\leq\tau_B}f_t(E[\bar{\rho}_t]) - \sum_{t\leq\tau_B}f_t^* = \sum_{t\leq\tau_B}f_t(\tilde{\rho}_t) - \sum_{t\leq\tau_B}f_t^*.$$

For any τ_B, the following terms could be bounded according to the propositions proved in Appendix G and H using online learning (i.e., both regret and fit are bounded sub-linearly):

$$\sum_{t\leq\tau_B}f_t(\tilde{\rho}_t) - \sum_{t\leq\tau_B}f_t^* \leq \tau_B^{\nu_1},$$
$$||\sum_{t\leq\tau_B}\boldsymbol{g}_t(\tilde{\rho}_t)|| \leq \tau_B^{\nu_2},$$

where both of ν_1 and ν_2 are constants and are less than 1. We then apply such results to fit and regret and have

$$reg_o = \sum_{t\leq\tau_B}f_t(\tilde{\rho}_t) - \sum_{t\leq\tau_B}f_t^* \leq \tau_B^{\nu_1} \leq \Xi_2(B)^{\nu_1} \triangleq \Omega_1,$$
$$fit_o \leq ||\sum_{t\leq\tau_B}\boldsymbol{g}_t(\tilde{\rho}_t)|| + \Gamma \leq \tau_B^{\nu_2} + \Gamma \leq \Xi_2(B)^{\nu_2} + \Gamma \triangleq \Omega_2,$$

where Ω_1 and Ω_2 are constants. Note that both Ω_1 and Ω_2 are at most sub-linear growth with respect to budget B. □

D. Proof of Lemma 4

Proof. According to the research [25], the regret is ensured based on UCB plays (i.e., the following inequality holds):

$$\begin{aligned} reg_b &\triangleq E_{\xi\sim\mathscr{D}}[\sum_{t\leq\tau_B}q_{e^*t} - \sum_{t\leq\tau_B,e}\bar{x}_{et}q_{et}] \\ &\leq \varrho_2 + \varrho_3 log(\varrho_1 + \varrho_0 B), \end{aligned}$$

where all of the ϱ are constants and q_{et} is the reward obtained by choosing edge e. Note that the objective of theirs is to maximize the overall reward obtained. As a result, the optimum is the subtrahend (i.e., the term $\sum_{t\leq\tau_B}q_{e^*t}$). However, in this paper, our objective is to minimize the overall latency. The relationship between the latency we concerned and the reward is illustrated as follows, by introducing a constant:

$$q_{et} = \chi - \psi_{et}, \quad \forall t \leq \tau_B, \forall e,$$

where χ is a positive constant to ensure the positive property of all rewards. Then, we have the following inequality:

$$E_{\xi\sim\mathscr{D}}[\sum_{t\leq\tau_B}(\chi - \psi_{e^*t}) - \sum_{t\leq\tau_B,e}\bar{x}_{et}(\chi - \psi_{et})] \leq \varrho_0 B - \Xi_1(B).$$

Note that we only need to choose one edge for global model aggregation per epoch. Thus, we have

$$\sum_{t\leq\tau_B}\chi = \sum_{t\leq\tau_B}\chi * 1 = \sum_{t\leq\tau_B}\chi * \sum_e \bar{x}_{et} = \sum_{t\leq\tau_B}\sum_e \bar{x}_{et}\chi.$$

After combing previous two inequalities together, we have

$$E_{\xi\sim\mathscr{D}}[\sum_{t\leq\tau_B}\sum_e\bar{x}_{et}\psi_{et} - \sum_{t\leq\tau_B}\psi_{e^*t}] \leq \varrho_0 B - \Xi_1(B) \triangleq \Omega_3,$$

which exactly matches the regret of our objective for bandit plays (i.e., the following inequality holds):

$$reg_b = E_{\xi\sim\mathscr{D}}[\sum_{t\leq\tau_B}\sum_e\bar{x}_{et}\psi_{et} - \sum_{t\leq\tau_B}\psi_{e^*t}] \leq \Omega_3.$$ □

E. Proof of Theorem 1

Proof. Lemma 1 confirms that if the global model aggregations reach to a desired value (i.e., $\tau_B \geq \mathcal{O}(\sum_{t\leq\tau_B} \bar{\rho}_t^{-\alpha}/\varepsilon)$), we have the following inequality in terms of the global loss:

$$E[\frac{\sum_t l_t}{\tau_B} - l_{opt}] \leq \frac{\sum_{i,t} u \max\{1, (\frac{2n_{it}}{M_t\bar{\rho}_t})^\alpha\}\mathcal{H}_{max}}{N\tau_B} \leq \varepsilon.$$

After that, Lemma 2 confirms the lower bound and the upper bound of the stopping time τ_B given the budget B, i.e.,

$$\tau_B \geq \Xi_1(B) \geq \sum_{t\leq\tau_B} \frac{\Xi_1(B)}{\Xi_2(B)}.$$

Then, we try to link the stopping time τ_B given the B and the desired global model aggregations $\mathcal{O}(\sum_{t\leq\tau_B} \bar{\rho}_t^{-\alpha}/\varepsilon)$. By using Lemma 3 and the definition of $\boldsymbol{g}$, we have

$$[E[\sum_{t\leq\tau_B}\{\frac{\varpi_t\bar{\rho}_t^{-\alpha}}{\varepsilon} - \frac{\Xi_1(B)}{\Xi_2(B)}\}]]^+ \leq \Xi_2(B)^{\nu_2} + \Gamma = \Omega_2,$$

since g_{2t} is also a dimension of constraint $\boldsymbol{g}_t$. It equals to

$$E[\sum_{t\leq\tau_B}\frac{\varpi_t\bar{\rho}_t^{-\alpha}}{\varepsilon}] - \Omega_2 \leq \sum_{t\leq\tau_B}\frac{\Xi_1(B)}{\Xi_2(B)},$$

since $[\cdot]^+ = \max\{\cdot, 0\} \geq (\cdot)$. We then apply the results from Lemma 2 into previous inequality and we have

$$E[\sum_{t\leq\tau_B}\frac{\varpi_t\bar{\rho}_t^{-\alpha}}{\varepsilon}] - \Omega_2 \leq \sum_{t\leq\tau_B}\frac{\Xi_1(B)}{\Xi_2(B)} \leq \Xi_1(B) \leq \tau_B.$$

Thus, from the perspective of the expectation, we have

$$\tau_B \geq \mathcal{O}(\sum_{t\leq\tau_B}\bar{\rho}_t^{-\alpha}/\varepsilon),$$

which implies the ensured the global model aggregations. □

F. Proof of Theorem 2

Proof. The objective of our problem $\mathbb{P}$ contains two decisions $\{\rho_t\}$ and $\{x_{et}\}$ (note that $\{\bar{\rho}_t\}$ and $\{\bar{x}_{et}\}$ are produced by our algorithm), and the input B. Its regret is

$$regret = E_{\xi\sim\mathscr{D}}[\bar{\mathcal{P}}_\xi - \mathcal{P}_\xi^*] = E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \bar{x}_{et}) - \mathcal{P}_\xi^*],$$

where $\bar{\rho}_t, \bar{x}_{et}$ are the decisions obtained from our designed algorithm and ρ_t^*, x_{et}^* are the optimum of $\mathbb{P}$. Note that the stopping time for such decisions is τ_B^*. We should also mention here that given B, $\tau_B^* \leq \tau_B$. Otherwise, the first τ_B terms of $\{x_{et}^*\}$ are the better choice, which results in the contradictory, since $\{x_{et}^*\}$ is actually the optimum. We show its upper bound through two steps. We first consider

$$\mathcal{R}_1 \triangleq E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \bar{x}_{et})] - E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \hat{x}_{et}^*)],$$

where $\hat{x}_{et}^*$ is the optimum under $\bar{\rho}_t$ and stopping time τ_B^*. This form can be re-written into the sum of following terms:

$$\begin{aligned}\mathcal{R}_1 = &E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \bar{x}_{et})] - E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \ddot{x}_{et}^*)]\\ &+E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \ddot{x}_{et}^*)] - E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \hat{x}_{et}^*)],\end{aligned}$$

where $\ddot{x}_{et}^*$ is the optimum given $\bar{\rho}_t$ and stopping time τ_B.

The first two terms in $\mathcal{R}_1$ actually obey the results directly from preliminary works [25] (i.e., given budget B, we have)

$$E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \bar{x}_{et}) - \mathcal{P}_\xi(\bar{\rho}_t, \ddot{x}_{et}^*)] \leq \mathcal{O}(\log B),$$

which obeys the same form mentioned in Lemma 4.

For the next two terms, since $\tau_B^* \leq \tau_B$, we can split the first term $E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \ddot{x}_{et}^*)]$ further into two parts (i.e., in the scope of $[1, \tau_B^*]$ and in the scope of $(\tau_B^*, \tau_B]$). In the scope of $[1, \tau_B^*]$, $\ddot{x}_{et}^* = \hat{x}_{et}^*$, otherwise, we can use the better one from $\{\ddot{x}_{et}^*\}$ and $\{\hat{x}_{et}^*\}$ to construct a more better choice. Thus, the last two terms in $\mathcal{R}_1$ is upper-bounded as follows:

$$E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \ddot{x}_{et}^*) - \mathcal{P}_\xi(\bar{\rho}_t, \hat{x}_{et}^*)] \leq (\tau_B - \tau_B^*)\max_{e,\xi\sim\mathscr{D}}\{\psi_{et}\}.$$

Also according to the previous work [25], we have

$$\tau_B - \tau_B^* \leq \mathcal{O}(\log\tau_B).$$

Then, $\mathcal{R}_1$ can be upper-bounded as follows:

$$\mathcal{R}_1 \leq \mathcal{O}(\log B) + \mathcal{O}(\log\tau_B) \leq \mathcal{O}(\log B) + \mathcal{O}(\log\tau_B).$$

After that, we consider the desired form needed as follows:

$$\mathcal{R}_2 \triangleq E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \bar{x}_{et})] - E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\rho_t^*, x_{et}^*)],$$

which can be also split into three parts (i.e., we have):

$$\begin{aligned}\mathcal{R}_2 = &E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \bar{x}_{et})] - E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \hat{x}_{et}^*)]\\ &+E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \hat{x}_{et}^*)] - E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\rho_t^*, \hat{x}_{et}^*)]\\ &+E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\rho_t^*, \hat{x}_{et}^*)] - E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\rho_t^*, x_{et}^*)].\end{aligned}$$

The first two terms in $\mathcal{R}_2$ is the result just mentioned (i.e., $\mathcal{R}_1$). The next two terms imply the difference on the objective under various ρ_t. Note that the difference on ρ_t is exactly the objective of our proposed subproblem $\mathbb{P}_1$. And Lemma 3 just proves the regret on the objective of $\mathbb{P}_1$ (i.e., we have):

$$E_{\xi\sim\mathscr{D}}[E[\sum_{t\leq\tau_B} f_t(\bar{\rho}_t)] - \sum_{t\leq\tau_B} f_t^*] \leq E_{\xi\sim\mathscr{D}}[\tau_B^{\nu_1}] = \tau_B^{\nu_1},$$

which implies the following inequality holds:

$$E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \hat{x}_{et}^*, \hat{B}^*) - \mathcal{P}_\xi(\rho_t^*, \hat{x}_{et}^*, \hat{B}^*)] \leq \tau_B^{\nu_1}.$$

And the last two terms in $\mathcal{R}_2$ are all considered in the scope of τ_B^*, given B and ρ_t^*. Actually, it equals to

$$E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\rho_t^*, \hat{x}_{et}^*) - \mathcal{P}_\xi(\rho_t^*, x_{et}^*)] = \sum_{t\leq\tau_B^*}(\hat{x}_{et}^* - x_{et}^*)\psi_{et}.$$

Since the terms regarding $\{\rho_t^*\}$ from these two objectives are the same, we can make a substitute within $[1, \tau_B^*]$ as

$$\mathcal{P}_\xi(\rho_t^*, \hat{x}_{et}^*) - \mathcal{P}_\xi(\rho_t^*, x_{et}^*) = \mathcal{P}_\xi(\bar{\rho}_t, \hat{x}_{et}^*) - \mathcal{P}_\xi(\bar{\rho}_t, x_{et}^*).$$

Similarly, upon the results mentioned before, we have

$$E_{\xi\sim\mathscr{D}}[\mathcal{P}_\xi(\bar{\rho}_t, \hat{x}_{et}^*) - \mathcal{P}_\xi(\bar{\rho}_t, x_{et}^*)] \leq \mathcal{O}(\log B),$$

where the scope considered is τ_B^* instead of τ_B.

Finally, combing all of these three results together, we have

$$\mathcal{R}_2 \leq \mathcal{O}(\log B) + \mathcal{O}(\log\tau_B) + \mathcal{O}(\tau_B^{\nu_1}) + \mathcal{O}(\log B),$$

where $\nu_1 < 1$. We should mention here that $\log B$ is upper-bounded by $\mathcal{O}(\log\tau_B)$ by using the relationship between τ_B and Ξ_1. Thus, it implies that the overall regret is

$$regret = E_{\xi\sim\mathscr{D}}[\bar{\mathcal{P}}_\xi - \mathcal{P}_\xi^*] \leq \mathcal{O}(\tau_B^{\nu_1} + \log\tau_B),$$

where all of these two terms are sub-linear respect to τ_B. □

G. Proof of Proposition 1

In this subsection, we show the detailed analysis on

$$||\sum_{t\leq\tau_B}\boldsymbol{g}_t(\tilde{\rho}_t)|| \leq \tau_B^{\nu_2},$$

where ν_2 is a constant and it is actually less than 1.

Proof. According to the update of $\boldsymbol{\lambda}$, we have

$$||\boldsymbol{\lambda}_{t+1}||^2 = ||[\boldsymbol{\lambda}_t + \gamma_2\boldsymbol{g}_t(\tilde{\rho})]^+||^2 \leq ||\boldsymbol{\lambda}_t + \gamma_2\boldsymbol{g}_t(\tilde{\rho})||^2,$$

where such function $[\cdot]^+$ used for each dimension essentially decreases the absolute value. Expanding the terms, we have

$$\Delta(\boldsymbol{\lambda}_t) \triangleq \frac{(||\boldsymbol{\lambda}_{t+1}||^2 - ||\boldsymbol{\lambda}_t||^2)}{2} \leq \gamma_2\boldsymbol{\lambda}_t^\top\boldsymbol{g}_t(\tilde{\rho}_t) + \frac{\gamma_2^2}{2}||\boldsymbol{g}_t(\tilde{\rho}_t)||^2.$$

Since $\tilde{\rho}_{t+1}$ is the optimum of $\mathbb{P}_{1,t}$, by using the interior point $\hat{\rho}_t$ mentioned in Assumption 3, we have

$$\begin{aligned}
&\nabla f_t(\tilde{\rho}_t)(\tilde{\rho}_{t+1} - \tilde{\rho}_t) + \boldsymbol{\lambda}_{t+1}^\top\boldsymbol{g}_t(\tilde{\rho}_{t+1}) + \frac{||\tilde{\rho}_{t+1} - \tilde{\rho}_t||^2}{2\gamma_1}\\
&\leq \nabla f_t(\tilde{\rho}_t)(\hat{\rho}_t - \tilde{\rho}_t) + \boldsymbol{\lambda}_{t+1}^\top\boldsymbol{g}_t(\hat{\rho}_t) + \frac{||\hat{\rho}_t - \tilde{\rho}_t||^2}{2\gamma_1}\\
&\leq \nabla f_t(\tilde{\rho}_t)(\hat{\rho}_t - \tilde{\rho}_t) - \varsigma\boldsymbol{\lambda}_{t+1}^\top\mathbf{1} + \frac{||\hat{\rho}_t - \tilde{\rho}_t||^2}{2\gamma_1}\\
&\leq \nabla f_t(\tilde{\rho}_t)(\hat{\rho}_t - \tilde{\rho}_t) - \varsigma||\boldsymbol{\lambda}_{t+1}|| + \frac{||\hat{\rho}_t - \tilde{\rho}_t||^2}{2\gamma_1},
\end{aligned}$$

where the second inequality sign holds due to Assumption 3 and the third inequality sign holds because $||\boldsymbol{\lambda}_{t+1}||$ is less or equal to $\boldsymbol{\lambda}_{t+1}^\top\mathbf{1}$ for all any non-negative vectors $\boldsymbol{\lambda}_{t+1}$. Then, we re-arrange all of these terms as follows:

$$\begin{aligned}
\boldsymbol{\lambda}_{t+1}^\top\boldsymbol{g}_t(\tilde{\rho}_{t+1}) &\leq \nabla f_t(\tilde{\rho}_t)(\tilde{\rho}_{t+1} - \tilde{\rho}_t) - \nabla f_t(\tilde{\rho}_t)(\hat{\rho}_t - \tilde{\rho}_t)\\
&\quad -\varsigma||\boldsymbol{\lambda}_{t+1}|| + \frac{||\hat{\rho}_t - \tilde{\rho}_t||^2 - ||\tilde{\rho}_{t+1} - \tilde{\rho}_t||^2}{2\gamma_1}\\
&\leq \nabla f_t(\tilde{\rho}_t)(\tilde{\rho}_{t+1} - \tilde{\rho}_t) - \nabla f_t(\tilde{\rho}_t)(\hat{\rho}_t - \tilde{\rho}_t) - \varsigma||\boldsymbol{\lambda}_{t+1}|| + \frac{G_d^2}{2\gamma_1}\\
&\leq \nabla f_t(\tilde{\rho}_t)\{(\tilde{\rho}_{t+1} - \tilde{\rho}_t) + (\hat{\rho}_t - \tilde{\rho}_t)\} - \varsigma||\boldsymbol{\lambda}_{t+1}|| + \frac{G_d^2}{2\gamma_1}\\
&\leq 2G_fG_d - \varsigma||\boldsymbol{\lambda}_{t+1}|| + \frac{G_d^2}{2\gamma_1} \quad\triangleq\quad \Phi_{t+1},
\end{aligned}$$

where the second inequality sign holds due to bounded domain and the fact that $||\tilde{\rho}_{t+1} - \tilde{\rho}_t||^2 \geq 0$; the third inequality sign holds due to the Cauchy-Schwartz Inequality; and the fourth inequality sign holds due to bounded gradient and domain. Next, we plug the previous inequality into the one regarding the relationship between two consecutive $\boldsymbol{\lambda}$ and have

$$\begin{aligned}
&\Delta(\boldsymbol{\lambda}_{t+1}) \leq \gamma_2\boldsymbol{\lambda}_{t+1}^\top\boldsymbol{g}_{t+1}(\tilde{\rho}_{t+1}) + \frac{\gamma_2^2}{2}||\boldsymbol{g}_{t+1}(\tilde{\rho}_{t+1})||^2\\
&\leq \gamma_2\boldsymbol{\lambda}_{t+1}^\top(\boldsymbol{g}_{t+1}(\tilde{\rho}_{t+1}) - \boldsymbol{g}_t(\tilde{\rho}_{t+1})) + \Phi_{t+1} + \frac{\gamma_2^2G_g^2}{2}\\
&\leq \gamma_2\boldsymbol{\lambda}_{t+1}^\top[\boldsymbol{g}_{t+1}(\tilde{\rho}_{t+1}) - \boldsymbol{g}_t(\tilde{\rho}_{t+1})]^+ + \Phi_{t+1} + \frac{\gamma_2^2G_g^2}{2}\\
&\leq \gamma_2||\boldsymbol{\lambda}_{t+1}||\hat{V}(\boldsymbol{g}) + \Phi_{t+1} + \frac{\gamma_2^2G_g^2}{2},
\end{aligned}$$

where the second inequality sign holds because we add two complementary terms (i.e., $\mp\gamma_2\boldsymbol{\lambda}_{t+1}^\top\boldsymbol{g}_t(\tilde{\rho}_{t+1})$); the third inequality sign holds due to the property of $[\cdot]^+$ and the non-negative property of $\boldsymbol{\lambda}$; the fourth inequality sign holds due to Assumption 3. After that, we show the following fact:

$$\forall t, ||\boldsymbol{\lambda}_t|| \leq ||\bar{\boldsymbol{\lambda}}|| \triangleq \gamma_2G_g + \frac{2G_fG_d + G_d^2/(2\gamma_1) + \gamma_2G_g^2/2}{\varsigma - \hat{V}(\boldsymbol{g})}.$$

We show the correctness of it by using the contradiction. Without loss of generality, we denote by $t+2$ the first time index that breaks the previous inequality, i.e.,

$$||\boldsymbol{\lambda}_{t+1}|| \leq ||\bar{\boldsymbol{\lambda}}|| \leq ||\boldsymbol{\lambda}_{t+2}||.$$

By using the update on $\boldsymbol{\lambda}_{t+1}$, we have

$$\begin{aligned}
&||\boldsymbol{\lambda}_{t+1}|| \geq ||\boldsymbol{\lambda}_{t+2}|| - ||\boldsymbol{\lambda}_{t+2} - \boldsymbol{\lambda}_{t+1}||\\
&= ||\boldsymbol{\lambda}_{t+2}|| - ||[\boldsymbol{\lambda}_{t+1} + \gamma_2\boldsymbol{g}_{t+1}(\tilde{\rho}_{t+1})]^+ - \boldsymbol{\lambda}_{t+1}||\\
&\geq ||\boldsymbol{\lambda}_{t+2}|| - ||\boldsymbol{\lambda}_{t+1} + \gamma_2\boldsymbol{g}_{t+1}(\tilde{\rho}_{t+1}) - \boldsymbol{\lambda}_{t+1}||\\
&\geq ||\boldsymbol{\lambda}_{t+2}|| - ||\gamma_2\boldsymbol{g}_{t+1}(\tilde{\rho}_{t+1})|| > ||\bar{\boldsymbol{\lambda}}|| - \gamma_2G_g,
\end{aligned}$$

where the first inequality holds due to the Triangle Inequality; the first equation sign holds due to the definition; the second inequality sign holds due to the property of $[\cdot]^+$; and the fourth inequality sign holds due to the hypothesis. After plugging the previous inequality into the inequality regarding $\Delta(\boldsymbol{\lambda}_{t+1})$, we have $\Delta(\boldsymbol{\lambda}_{t+1})$ is less than 0, leading to $||\boldsymbol{\lambda}_{t+2}|| < ||\boldsymbol{\lambda}_{t+1}||$, which contradicts the hypothesis. Thus, the fact is proved.

We further use the update on $\boldsymbol{\lambda}$ and have

$$[\boldsymbol{\lambda}_\tau + \gamma_2\boldsymbol{g}_{\tau_B}(\tilde{\rho}_{\tau_B})]^+ \geq ... \geq \boldsymbol{\lambda}_1 + \sum\nolimits_{t\leq\tau_B}\gamma_2\boldsymbol{g}_t(\tilde{\rho}_t).$$

Since $\boldsymbol{\lambda}_1 = \mathbf{0}$, re-arranging the previous inequality, we have

$$\sum_{t\leq\tau_B}\boldsymbol{g}_t(\tilde{\rho}_t) \leq \frac{\boldsymbol{\lambda}_{\tau_B+1}}{\gamma_2} - \frac{\boldsymbol{\lambda}_1}{\gamma_2} \leq \frac{\boldsymbol{\lambda}_{\tau_B+1}}{\gamma_2}.$$

Thus, for the fit, we have the following inequality:

$$||\sum_{t\leq\tau_B}\boldsymbol{g}_t(\tilde{\rho}_t)|| \leq ||\frac{\boldsymbol{\lambda}_{\tau_B+1}}{\gamma_2}|| \leq \frac{||\bar{\boldsymbol{\lambda}}||}{\gamma_2}.$$ □

H. Proof of Proposition 2

In this subsection, we show the detailed analysis on

$$\sum_{t\leq\tau_B}f_t(\tilde{\rho}_t) - \sum_{t\leq\tau_B}f_t^* \leq \tau_B^{\nu_1},$$

where ν_1 is a constant and it is also less than 1.

Proof. The objective of $\mathbb{P}_{1,t}$ implies that it is a $1/\gamma_1$-strongly convex function with respect to the variable $\tilde{\rho}$ (i.e., $\forall\zeta_1,\zeta_2$):

$$\mathcal{P}_{1,t}(\zeta_1) \geq \mathcal{P}_{1,t}(\zeta_2)(\zeta_2 - \zeta_1) + \frac{||\zeta_2 - \zeta_1||^2}{2\gamma_1}.$$

Note that $\tilde{\rho}_{t+1}$ solved is exactly the optimum for $\mathcal{P}_{1,t}$, and it also holds the optimality condition as follows:

$$\nabla\mathcal{P}_{1,t}(\tilde{\rho}_{t+1})(\tilde{\rho}_t^* - \tilde{\rho}_{t+1}) \geq 0,$$

where $\tilde{\rho}_t^*$ is the optimum for the objective of $\mathbb{P}_1$ per time (i.e., f_t). By setting $\zeta_1 = \tilde{\rho}_{t+1}$, $\zeta_2 = \tilde{\rho}_t^*$, we have

$$\mathcal{P}_{1,t}(\tilde{\rho}_t^*) \geq \mathcal{P}_{1,t}(\tilde{\rho}_{t+1}) + \frac{1}{2\gamma_1}||\tilde{\rho}_t^* - \tilde{\rho}_{t+1}||^2.$$

After adding $f_t(\tilde{\rho}_t)$ on both of two sides for the previous inequality, expanding related objective according to its definition and using the property of a convex function $f_t(\cdot)$ (i.e., $f_t(\tilde{\rho}_t^*) \geq f_t(\tilde{\rho}_t) + \nabla f_t(\tilde{\rho}_t)(\tilde{\rho}_t^* - \tilde{\rho}_t)$), we have

$$\begin{aligned}
&f_t(\tilde{\rho}_t) + \nabla f_t(\tilde{\rho}_t)(\tilde{\rho}_{t+1} - \tilde{\rho}_t) + \boldsymbol{\lambda}_{t+1}^\top \boldsymbol{g}_t(\tilde{\rho}_{t+1}) + \frac{||\tilde{\rho}_{t+1} - \tilde{\rho}_t||^2}{2\gamma_1} \\
&\leq f_t(\tilde{\rho}_t^*) + \boldsymbol{\lambda}_{t+1}^\top \boldsymbol{g}_t(\tilde{\rho}_t^*) + \frac{||\tilde{\rho}_t^* - \tilde{\rho}_t||^2}{2\gamma_1} - \frac{||\tilde{\rho}_t^* - \tilde{\rho}_{t+1}||^2}{2\gamma_1} \\
&\leq f_t(\tilde{\rho}_t^*) + \frac{||\tilde{\rho}_t^* - \tilde{\rho}_t||^2}{2\gamma_1} - \frac{||\tilde{\rho}_t^* - \tilde{\rho}_{t+1}||^2}{2\gamma_1},
\end{aligned}$$

where the second inequality holds due to the fact $\boldsymbol{g}_t(\tilde{\rho}_t^*) \preceq \mathbf{0}$. Note that the optimum is also a feasible solution and it makes the value of $\boldsymbol{g}$ negative. We then study the gradient term:

$$\begin{aligned}
&-\nabla f_t(\tilde{\rho}_t)(\tilde{\rho}_{t+1} - \tilde{\rho}_t) \leq ||\nabla f_t(\tilde{\rho}_t)||\ ||\tilde{\rho}_{t+1} - \tilde{\rho}_t|| \\
&\leq \frac{||\nabla f_t(\tilde{\rho}_t)||^2}{2\zeta} + \frac{\zeta}{2}||\tilde{\rho}_{t+1} - \tilde{\rho}_t|| \leq \frac{G_f^2}{2\zeta} + \frac{\zeta}{2}||\tilde{\rho}_{t+1} - \tilde{\rho}_t||,
\end{aligned}$$

where ζ is an arbitrary positive constant, the first inequality sign holds due to the property of norms; the second inequality sign holds due to basic algebra formula; and the third inequality holds due to bounded gradient. We then have

$$\begin{aligned}
&f_t(\tilde{\rho}_t) + \boldsymbol{\lambda}_{t+1}^\top \boldsymbol{g}_t(\tilde{\rho}_{t+1}) \leq f_t(\tilde{\rho}_t^*) + (\frac{\zeta}{2} - \frac{1}{2\gamma_1})||\tilde{\rho}_{t+1} - \tilde{\rho}_t||^2 \\
&\quad + \frac{1}{2\gamma_1}(||\tilde{\rho}_t^* - \tilde{\rho}_t||^2 - ||\tilde{\rho}_t^* - \tilde{\rho}_{t+1}||^2) + \frac{G_f^2}{2\zeta} \\
&= f_t(\tilde{\rho}_t^*) + \frac{1}{2\gamma_1}(||\tilde{\rho}_t^* - \tilde{\rho}_t||^2 - ||\tilde{\rho}_t^* - \tilde{\rho}_{t+1}||^2) + \frac{\gamma_1 G_f^2}{2},
\end{aligned}$$

where the first equation sign holds because we set $\zeta = 1/\gamma_1$. Next, by using the results from previous proposition, we have

$$\begin{aligned}
&\frac{\Delta(\boldsymbol{\lambda}_{t+1})}{\gamma_2} + f_t(\tilde{\rho}_t) \leq \boldsymbol{\lambda}_{t+1}^\top \boldsymbol{g}_{t+1}(\tilde{\rho}_{t+1}) + \frac{\gamma_2}{2}||\boldsymbol{g}_{t+1}(\tilde{\rho}_{t+1})||^2 \\
&\qquad + f_t(\tilde{\rho}_t) + \boldsymbol{\lambda}_{t+1}^\top \boldsymbol{g}_t(\tilde{\rho}_{t+1}) - \boldsymbol{\lambda}_{t+1}^\top \boldsymbol{g}_t(\tilde{\rho}_{t+1}) \\
&= f_t(\tilde{\rho}_t) + \boldsymbol{\lambda}_{t+1}^\top \boldsymbol{g}_{t+1}(\tilde{\rho}_{t+1}) \\
&\qquad + \frac{\gamma_2}{2}||\boldsymbol{g}_{t+1}(\tilde{\rho}_{t+1})||^2 + \boldsymbol{\lambda}_{t+1}^\top(\boldsymbol{g}_t(\tilde{\rho}_{t+1}) - \boldsymbol{g}_t(\tilde{\rho}_{t+1})) \\
&\leq f_t(\tilde{\rho}_t^*) + \frac{1}{2\gamma_1}(||\tilde{\rho}_t^* - \tilde{\rho}_t||^2 - ||\tilde{\rho}_t^* - \tilde{\rho}_{t+1}||^2) + \gamma_1 G_f^2/2 \\
&\qquad + \gamma_2 G_g^2/2 + ||\boldsymbol{\lambda}_{t+1}||\ [\boldsymbol{g}_t(\tilde{\rho}_{t+1}) - \boldsymbol{g}_t(\tilde{\rho}_{t+1})]^+,
\end{aligned}$$

where the first inequality sign holds due to previous proved results and two complementary terms (i.e., $\pm\boldsymbol{\lambda}_{t+1}^\top \boldsymbol{g}_t(\tilde{\rho}_{t+1})$); the first equation sign holds due to re-arrangement of terms; and the second inequality sign holds also due to already proved results, bounded gradient and the property of $[\cdot]^+$. After that, we study those intermediate terms as follows:

$$\begin{aligned}
&||\tilde{\rho}_t^* - \tilde{\rho}_t||^2 = ||\tilde{\rho}_t^* - \tilde{\rho}_t||^2 - ||\tilde{\rho}_t - \tilde{\rho}_{t-1}^*||^2 + ||\tilde{\rho}_t - \tilde{\rho}_{t-1}^*||^2 \\
&= ||\tilde{\rho}_t^* - \tilde{\rho}_{t-1}^*||\ ||\tilde{\rho}_t^* - 2\tilde{\rho}_t + \tilde{\rho}_{t-1}^*|| + ||\tilde{\rho}_t - \tilde{\rho}_{t-1}^*||^2 \\
&\leq 2G_d||\tilde{\rho}_t^* - \tilde{\rho}_{t-1}^*|| + ||\tilde{\rho}_t - \tilde{\rho}_{t-1}^*||^2,
\end{aligned}$$

where the first equation sign holds due to two complementary terms (i.e., $\mp||\tilde{\rho}_t - \tilde{\rho}_{t-1}^*||^2$); the second equation sign holds due to the difference of two squares; and the first inequality sign holds due to the Triangle Inequality and the bounded domain. Combing previous two inequalities together, we have

$$\begin{aligned}
&\frac{\Delta(\boldsymbol{\lambda}_{t+1})}{\gamma_2} + f_t(\tilde{\rho}_t) \leq f_t(\tilde{\rho}_t^*) + ||\boldsymbol{\lambda}_{t+1}||\ \hat{V}(\boldsymbol{g}_t) + \frac{\gamma_2 G_g^2}{2} + \frac{\gamma_1 G_f^2}{2} \\
&+ \frac{1}{2\gamma_1}(2G_d||\tilde{\rho}_t^* - \tilde{\rho}_{t-1}^*|| + ||\tilde{\rho}_t - \tilde{\rho}_{t-1}^*||^2 - ||\tilde{\rho}_t^* - \tilde{\rho}_{t+1}||^2),
\end{aligned}$$

where $\hat{V}(\boldsymbol{g}_t) \triangleq [\boldsymbol{g}_t(\tilde{\rho}_{t+1}) - \boldsymbol{g}_t(\tilde{\rho}_{t+1})]^+$, similar to $\hat{V}(\boldsymbol{g})$. We sum up all of these terms from $t = 1$ to τ_B, we have

$$\begin{aligned}
&\sum_{t\leq\tau_B}\{\frac{\Delta(\boldsymbol{\lambda}_{t+1})}{\gamma_2} + f_t(\tilde{\rho}_t)\} \leq \sum_{t\leq\tau_B} f_t(\tilde{\rho}_t^*) + \frac{\gamma_2 G_g^2 \tau_B}{2} + \frac{\gamma_1 G_f^2 \tau_B}{2} \\
&\quad + \sum_{t\leq\tau_B}||\boldsymbol{\lambda}_{t+1}||\ \hat{V}(\boldsymbol{g}_t) + \sum_{t\leq\tau_B}\frac{G_d}{\gamma_1}||\tilde{\rho}_t^* - \tilde{\rho}_{t-1}^*|| \\
&\quad + \sum_{t\leq\tau_B}\frac{1}{2\gamma_1}(||\tilde{\rho}_t - \tilde{\rho}_{t-1}^*||^2 - ||\tilde{\rho}_t^* - \tilde{\rho}_{t+1}||^2).
\end{aligned}$$

For simplicity, we define $\hat{V}_1 \triangleq \sum_{t\leq\tau_B}||\tilde{\rho}_t^* - \tilde{\rho}_{t-1}^*||$ as well as $\hat{V}_2 \triangleq \sum_{t\leq\tau_B}\max_t \hat{V}(\boldsymbol{g}_t)$. Then, we have

$$\begin{aligned}
&\sum_{t\leq\tau_B}\{\frac{\Delta(\boldsymbol{\lambda}_{t+1})}{\gamma_2} + f_t(\tilde{\rho}_t)\} \leq \sum_{t\leq\tau_B} f_t(\tilde{\rho}_t^*) + \frac{\gamma_2 G_g^2 \tau_B}{2} + \frac{\gamma_1 G_f^2 \tau_B}{2} \\
&+ ||\bar{\boldsymbol{\lambda}}||\hat{V}_2 + \frac{G_d}{\gamma_1}\hat{V}_1 + \frac{1}{2\gamma_1}(||\tilde{\rho}_1 - \tilde{\rho}_0^*||^2 - ||\tilde{\rho}_{\tau_B}^* - \tilde{\rho}_{\tau_B+1}||^2).
\end{aligned}$$

At last, we focus on the regret, i.e.,

$$\begin{aligned}
&\sum_{t\leq\tau_B} f_t(\tilde{\rho}_t) - \sum_{t\leq\tau_B} f_t^* \leq \frac{\gamma_2 G_g^2 \tau_B}{2} + \frac{\gamma_1 G_f^2 \tau_B}{2} + ||\bar{\boldsymbol{\lambda}}||\hat{V}_2 \\
&\quad + \frac{G_d}{\gamma_1}\hat{V}_1 + \frac{||\tilde{\rho}_1 - \tilde{\rho}_0^*||^2}{2\gamma_1} - \frac{||\boldsymbol{\lambda}_{t+2}||^2}{2\gamma_2} + \frac{||\boldsymbol{\lambda}_2||^2}{2\gamma_2}.
\end{aligned}$$

Note that $||\tilde{\rho}_1 - \tilde{\rho}_0^*||^2$ is bounded by G_d^2, $||\boldsymbol{\lambda}_{t+2}||^2 \geq 0$ and $||\boldsymbol{\lambda}_2||^2 \leq \gamma_2^2 G_g^2$ if $\boldsymbol{\lambda}_1 = \mathbf{0}$. Thus, we have the regret:

$$\leq \frac{\gamma_2 G_g^2(\tau_B + 1)}{2} + \frac{\gamma_1 G_f^2 \tau_B}{2} + ||\bar{\boldsymbol{\lambda}}||\hat{V}_2 + \frac{G_d}{\gamma_1}\hat{V}_1 + \frac{G_d^2}{2\gamma_2}.$$

We choose proper step sizes as follows:

$$\gamma_1 = \gamma_2 = \max\{\sqrt{\hat{V}_1/\tau_B}, \sqrt{\hat{V}_2/\tau_B}\},$$

which balances the parts considered in the objective of $\mathbb{P}_{1,t}$. As a result, the regret is re-written under proper step sizes:

$$\sum_{t\leq\tau_B} f_t(\tilde{\rho}_t) - \sum_{t\leq\tau_B} f_t^* \leq \mathcal{O}(\max\{\sqrt{\hat{V}_1\tau_B}, \sqrt{\hat{V}_2\tau_B}\}).$$

Following this corollary, if the step sizes obey

$$\gamma_1 = \gamma_2 = \mathcal{O}(\tau_B^{-1/3}),$$

the regret is further bounded by the following inequality:

$$\sum_{t\leq\tau_B} f_t(\tilde{\rho}_t) - \sum_{t\leq\tau_B} f_t^* \leq \mathcal{O}(\max\{\hat{V}_1\tau_B^{1/3}, \hat{V}_2\tau_B^{1/3}, \tau_B^{2/3}\}).$$

Since $\hat{V}_1$ and $\hat{V}_2$ are fixed for given system, the regret is actually sub-linearly growth as time goes, i.e.,

$$\sum_{t\leq\tau_B} f_t(\tilde{\rho}_t) - \sum_{t\leq\tau_B} f_t^* \leq \mathcal{O}(\tau_B^{\nu_1}),$$

where ν_1 is a constant and is less than 1. $\square$

References

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *PMLR International Conference on Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[2] W. House, "Consumer data privacy in a networked world: A framework for protecting privacy and promoting innovation in the global digital economy," *White House, Washington, DC*, pp. 1–62, 2012.

[3] "AWS Spot Instance," https://www.cloudomatic.com/awsspot-instance-price-analysis-for-last-3-months-sydney-region/, 2017.

[4] C.-C. Hung, G. Ananthanarayanan, L. Golubchik, M. Yu, and M. Zhang, "Wide-area analytics with multiple resources," in *ACM 13th EuroSys Conference*, 2018, pp. 1–16.

[5] E. Kurdoglu, Y. Liu, Y. Wang, Y. Shi, C. Gu, and J. Lyu, "Real-time bandwidth prediction and rate adaptation for video calls over cellular networks," in *ACM 7th International Conference on Multimedia Systems*, 2016, pp. 1–11.

[6] K. Cai, X. Liu, Y.-Z. J. Chen, and J. C. Lui, "An online learning approach to network application optimization with guarantee," in *IEEE Conference on Computer Communications*, 2018, pp. 2006–2014.

[7] S. Shalev-Shwartz and T. Zhang, "Stochastic dual coordinate ascent methods for regularized loss minimization," *Journal of Machine Learning Research*, vol. 14, pp. 567–599, 2013.

[8] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," *27th Annual Conference on Neural Information Processing Systems*, vol. 26, pp. 315–323, 2013.

[9] A. Defazio, F. Bach, and S. Lacoste-Julien, "Saga: A fast incremental gradient method with support for non-strongly convex composite objectives," *28th Annual Conference on Neural Information Processing Systems*, vol. 27, pp. 1646–1654, 2014.

[10] H. Daneshmand, A. Lucchi, and T. Hofmann, "Starting small-learning with adaptive sample sizes," in *33rd International Conference on Machine Learning*, 2016, pp. 1463–1471.

[11] E. Jothimurugesan, A. Tahmasbi, P. Gibbons, and S. Tirthapura, "Variance-reduced stochastic gradient descent on streaming data," in *32nd Annual Conference on Neural Information Processing Systems*, 2018, pp. 9906–9915.

[12] Y. Tu, Y. Ruan, S. Wang, S. Wagle, C. G. Brinton, and C. Joe-Wang, "Network-aware optimization of distributed learning for fog computing," in *IEEE Conference on Computer Communications*, 2020, pp. 2509–2518.

[13] Y. Jin, L. Jiao, Z. Qian, S. Zhang, S. Lu, and X. Wang, "Resource-efficient and convergence-preserving online participant selection in federated learning," in *40th IEEE International Conference on Distributed Computing Systems*, 2020, pp. 606–616.

[14] B. Luo, X. Li, S. Wang, J. Huang, and L. Tassiulas, "Cost-effective federated learning design," *arXiv:2012.08336*, pp. 1–10, 2020.

[15] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, pp. 2031–2063, 2020.

[16] N. H. Tran, W. Bao, A. Zomaya, N. M. NH, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *IEEE Conference on Computer Communications*, 2019, pp. 1387–1395.

[17] A. Asuncion and D. Newman, "Uci machine learning repository," https://archive.ics.uci.edu/ml/index.php, 2007.

[18] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "Rcv1: A new benchmark collection for text categorization research," *Journal of Machine Learning Research*, vol. 5, pp. 361–397, 2004.

[19] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems*, vol. 5, no. 4, pp. 1–19, 2015.

[20] Y. Jin, Z. Qian, S. Guo, S. Zhang, X. Wang, and S. Lu, "Ran-GJS: Orchestrating data analytics for heterogeneous geo-distributed edges," in *ACM 47th International Conference on Parallel Processing*, 2018, pp. 1–10.

[21] Y. Xu, C. Yu, J. Li, and Y. Liu, "Video telephony for end-consumers: measurement study of google+, ichat and skype," *IEEE/ACM Transactions on Networking*, vol. 22, no. 3, pp. 826–839, 2014.

[22] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "When edge meets learning: Adaptive control for resource-constrained distributed machine learning," in *IEEE Conference on Computer Communications*, 2018, pp. 63–71.

[23] M. Curran, M. S. Rahman, H. Gupta, and V. Sekar, "Rethinking virtual network embedding in reconfigurable networks," in *15th Annual IEEE International Conference on Sensing, Communication, and Networking*, 2018, pp. 1–9.

[24] A. Saha, N. Ganguly, S. Chakraborty, and A. De, "Learning network traffic dynamics using temporal point process," in *IEEE Conference on Computer Communications*, 2019, pp. 1927–1935.

[25] D. Zhou and C. Tomlin, "Budget-constrained multi-armed bandits with multiple plays," in *32nd AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018, pp. 4572–4579.

[26] V. Jalaparti, I. Bliznets, S. Kandula, B. Lucier, and I. Menache, "Dynamic pricing and traffic engineering for timely inter-datacenter transfers," in *ACM SIGCOMM Conference*, 2016, pp. 73–86.

[27] D. P. Bertsekas, "Incremental gradient, subgradient, and proximal methods for convex optimization: A survey," *Optimization for Machine Learning*, vol. 2010, no. 3, pp. 1–38, 2011.

[28] N. Roux, M. Schmidt, and F. Bach, "A stochastic gradient method with an exponential convergence _rate for finite training sets," *26th Annual Conference on Neural Information Processing Systems*, vol. 25, pp. 2663–2671, 2012.

[29] C. Ma, J. Konečný, M. Jaggi, V. Smith, M. I. Jordan, P. Richtárik, and M. Takáč, "Distributed optimization with arbitrary local solvers," *Taylor & Francis Optimization Methods and Software*, vol. 32, no. 4, pp. 813–848, 2017.

[30] X. Gao, M. Sitharam, and A. E. Roitberg, "Bounds on the jensen gap, and implications for mean-concentrated distributions," *arXiv:1712.05267*, pp. 1–16, 2017.

Yibo Jin received the BS degree from the Department of Computer Science and Technology, Nanjing University in 2017, where he is currently pursuing the PhD degree under the supervision of Professor Sanglu Lu. He was a visiting student with the Hong Kong Polytechnic University, Hong Kong in 2017. To date, he has already published over 15 papers, including in journals such as TPDS, and in conferences such as INFOCOM, ICDCS, ICPP, SECON and IWQoS. He received the Best Paper Candidates from WoWMoM 2021. His research interests include big data analytics and edge computing. He is a student member of the IEEE.

Lei Jiao received the Ph.D. degree in computer science from the University of Göttingen, Germany. He is currently an assistant professor at the Department of Computer and Information Science, University of Oregon, USA. Previously he worked as a member of technical staff at Alcatel-Lucent/Nokia Bell Labs in Dublin, Ireland and also as a researcher at IBM Research in Beijing, China. He is interested in the mathematics of optimization, control, learning, and mechanism design applied to computer and telecommunication systems, networks, and services. He publishes papers in journals such as JSAC, ToN, TPDS, TMC, and TDSC, and in conferences such as INFOCOM, MOBIHOC, ICNP, ICDCS, SECON, and IPDPS. He is a recipient of the NSF CAREER Award. He also received the Best Paper Awards of IEEE LANMAN 2013 and IEEE CNS 2019, and the 2016 Alcatel-Lucent Bell Labs UK and Ireland Recognition Award. He was on the program committees of conferences including INFOCOM, MOBIHOC, ICDCS, IWQoS, and ICC, and was also the program chair of multiple workshops with INFOCOM and ICDCS.

Zhuzhong Qian is a professor at the Department of Computer Science and Technology, and member of National Key Laboratory for Novel Software Technology, Nanjing University, P. R. China. He received his PhD. Degree in computer science at Nanjing University in 2007. Currently, his research interests include cloud computing, edge computing, and distributed machine learning. He is the chief member of several national research projects on cloud computing and edge computing. His research has been published in journals such as TPDS, TON, TC, and TMC, and in conferences such as INFOCOM, ICDCS, SECON, and IPDPS. He received best paper awards from ICA3PP 2014 and APNet 2018, and best paper candidates from WoWMoM 2021.

Sheng Zhang is an associate professor at the Department of Computer Science and Technology, Nanjing University. He is also a member of the State Key Lab. for Novel Software Technology. He received the BS and PhD degrees from Nanjing University in 2008 and 2014, respectively. His research interests include cloud computing and edge computing. To date, he has published more than 80 papers, including those appeared in TMC, TON, TPDS, TC, MobiHoc, ICDCS, INFOCOM, SECON, IWQoS and ICPP. He received the Best Paper Award of IEEE ICCCN 2020 and the Best Paper Runner-Up Award of IEEE MASS 2012. He is the recipient of the 2015 ACM China Doctoral Dissertation Nomination Award. He is a member of the IEEE and a senior member of the CCF.

Sanglu Lu received her BS, MS and PhD degrees from Nanjing University in 1992, 1995, and 1997, respectively, all in computer science. She is currently a professor in the Department of Computer Science and Technology and the State Key Laboratory for Novel Software Technology. Her research interests include distributed computing, wireless networks, and pervasive computing. She has published over 80 papers in referred journals and conferences in the above areas. She is a member of the IEEE.