Scheduling In-Band Network Telemetry with Convergence-Preserving Federated Learning

Yibo Jin, *Member, IEEE,* Lei Jiao, *Member, IEEE,* Mingtao Ji, *Member, IEEE,* Zhuzhong Qian, *Member, IEEE,* Sheng Zhang, *Member, IEEE,* Ning Chen, *Member, IEEE,* and Sanglu Lu, *Member, IEEE*

***Abstract*—Conducting federated learning across distributed sites with In-Band Network Telemetry (INT) based data collection faces critical challenges, including control decisions of different frequencies, convergence of the models being trained, and resource provisioning coupled over time. To study this problem, we formulate a non-linear mixed-integer program to optimize the long-term INT overhead, resource cost, and federated learning cost. We then design polynomial-time online algorithms to solve this problem with only observable inputs on the fly, featuring laziness-aware resource adaption, online-learning-based INT flow selection and model aggregation control, as well as expectation-preserving randomized dependent rounding. We rigorously prove the parameterized-constant competitive ratio of our approach against the offline optimum, and the time-averaged constraint violation that vanishes in the long run. With extensive trace-driven evaluations, we confirm the superiority of our approach over other alternative approaches for reducing total cost and the efficacy of our trained models for solving real machine learning problems, reducing the real-time cost by 34% on average.**

***Index Terms*—Online Provisioning, In-Band Network Telemetry, Federated Learning, Multi-timescale Optimization**

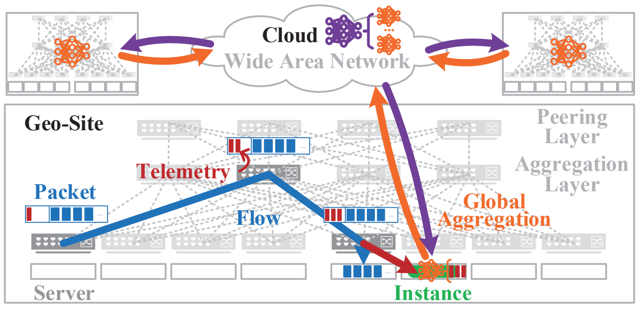


Fig. 1: Scenario of INT with Federated Learning

TABLE I: Mismatch of Time Scales (Testbed Results)

	Duration[1]	Size	Repetitions
INT	∼85ms	∼2.3KB	∼100 pps
Federated Learning	Minutes ∼ Hours	MB ∼ GB	Hundreds

1. The duration and the size here are investigated for one round of INT and for one round of model aggregation of federated learning.

I. Introduction

In-Band Network Telemetry (INT) [1, 2] enables the network switch to insert its state information (e.g., queue length, hop latency, link utilization) into the packet header. Such state data are then transported by the packet and separated from the packet payload for further processing at the destination. Compared to conventional network monitoring approaches, INT can collect data at line speed in the data-forwarding plane, measure desired switches en route, adapt to almost any encapsulation format, and scale to large networks of diverse types. P4 switch is an industrial example to realize INT [2].

INT data collected from networks are often aggregated [3] for data analytics [3–6]; however, this approach cannot work when one tries to aggregate INT data to a central place from multiple distributed sites in order to train machine learning models to solve complex problems. This is because each site may be owned and managed by a different operator or Internet Service Provider (ISP) [7]. Due to privacy concerns [8–10], a site may be unwilling to share or upload the measurement data of its internal networks to train machine learning models. Federated learning [11] can actually address this issue, where only the models being trained, rather than the raw training data, are exchanged between sites and can also save wide-area network (WAN) traffic. This scenario is visualized in Fig. 1.

Unfortunately, it is non-trivial to make INT and federated learning work together, which entails the joint orchestration of INT scheduling, resource provisioning, and federated learning control (e.g., convergence of the trained models). Operating such a system optimally faces multiple challenges as follows:

First, INT and federated learning often need to be managed at different frequencies in an online manner. INT is often conducted repeatedly in multiple rounds to capture the time-varying network states, and each round of INT can finish relatively fast even for the entire network of a site and needs to be managed in real time. In contrast, federated learning, upon required computing resources (e.g., virtual machines), often takes time [12, 13] and can only be adjusted less frequently. This misalignment, exemplified in Table 1, indicates that when setting up the resources for federated learning the amount of INT data to be learnt from is uncertain; once the resources are allocated, we cannot adjust such resources until sometime later—this inflexibility makes it hard to manage the system.

Second, both INT and federated learning desire long-term performance guarantees. While we may choose to conduct INT only if the data from adjacent rounds tend to have adequate difference, we are still subject to the total amount of resources available for data processing before it can be adjusted next

Manuscript received March 29, 2022; revised December 14, 2022; accepted February 13, 2023. Date of publication XXX, XX, 2023; date of current version February 26, 2023. This work is supported in part by the National Natural Science Foundation of China under grants No. 61832005, No. 61872175, and No. 61832008, in part by the China University Industry Research Innovation Foundation under grant No. 2021FNA04005, in part by the Fundamental Research Funds for the Central Universities under grant No. 2022300297, in part by the Collaborative Innovation Center of Novel Software Technology and Industrialization, and in part by the U.S. National Science Foundation under Grants CNS-2047719 and CNS-2225949. *(Corresponding authors: Lei Jiao; Zhuzhong Qian; Sheng Zhang)*

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

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

Digital Object Identifier xx.xxxx/xxxx.2023.xxxxxxx

time. For federated learning, to achieve desired convergence of the model being trained, we need to ensure the cumulative number of model aggregations over time, to exceed a pre-specified threshold. Such long-term constraints are generally not easy to meet in the online setting, since any current decisions will restrict the decisions that could be made in the future and may only turn out to be suboptimal.

Third, control decisions are time-coupled, or need to be made before observing their impact on the cost to be optimized. Changing the amount of resources incurs “switching cost” in terms of leading time, performance oscillation, and hardware wear-and-tear. Every current resource decision serves as a base for the switching cost that will be incurred if next time a different resource decision is made; yet, as the next resource decision is unknown now, it is not straightforward to determine the current resource allocation for optimizing the switching cost. INT decisions also need to be made without knowing the resultant measurements. Such an structure of the online decisions escalates the difficulty in the joint orchestration of both INT and federated learning in the system.

Existing research falls insufficient for addressing aforementioned challenges. Some [1–3, 14–16] have studied INT in terms of path selection, congestion control, and network visualization, without utilizing INT for sophisticated analytics and machine learning. Others [12, 17–21] have focused on the optimization of federated learning, but often do not consider training data collection, not to mention via INT. The rest [7, 10, 22–25] on geo-distributed analytics and machine learning have never captured federated learning or INT.

In this paper, we first model and formulate the optimization problem of convergence-preserving federated learning with INT-based data collection across geo-distributed sites. Seeking to optimize the long-term total cost of INT overhead, resource cost (i.e., operational and switching cost), and federated learning cost, we control flow selections and resource provisioning in each site as well as model aggregations across sites. Our formulation features control decisions at different frequencies, network-wise switch covering, and convergence of the model. Our problem is a non-linear mixed-integer program.

Afterwards, we design a group of polynomial-time algorithms that work together to solve this problem in an online manner. Our Algorithm 1 overcomes the time-coupling between decisions through postponing the adaption of the resources provisioned unless the cumulative non-switching cost since the last switch operation exceeds a controllable parameter of “laziness” times the switching cost incurred by the last switch operation. Invoked by Algorithm 1, our Algorithm 2 exploits a convex-concave problem transformation and the primal-dual-based updates to “learn” from previous decisions in order to dynamically allocate resources for INT processing and federated learning (i.e., “outer” learning) and select flows to collect the INT data (i.e., “inner” learning) without relying on future uncertainties, while addressing long-term resource caps and model aggregation requirements. Also invoked by Algorithm 1, our Algorithm 3 is responsible for converting the fractional decisions into integers via a randomized dependent rounding strategy, so that fractions are rounded in pairs while ensuring that the expectation of the integer equals the corresponding fraction.

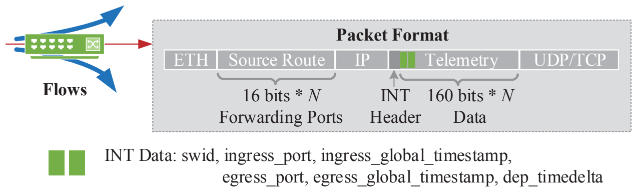

Flows
Packet Format
ETH
Source Route
IP
Telemetry
UDP/TCP
16 bits * N
Forwarding Ports
INT
Header
160 bits * N
Data
INT Data: swid, ingress_port, ingress_global_timestamp, egress_port, egress_global_timestamp, dep_timedelta

Fig. 2: Implementation of In-Band Network Telemetry

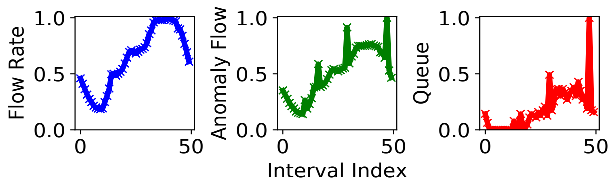
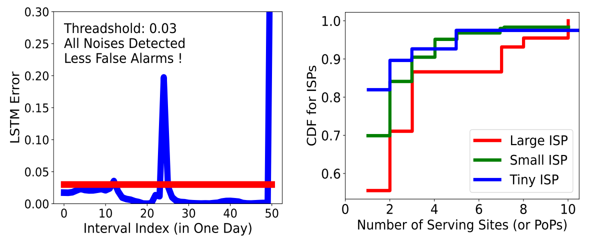


Fig. 3: Normal Flows, Flows with Noises, and Queue Length



(a) Detection Results (b) Serving Sites

Fig. 4: Telemetry for Queue Length Anomaly Detection

Further, we conduct rigorous analysis for our algorithms. We prove that our online approach can lead to a parameterized constant “competitive ratio” for the overall cost against the offline optimum where the inputs are observed at once at hindsight. We also prove that both the “regret” for the cumulative non-switching cost against its offline optimum and the cumulative violation of the long-term constraints, including the model quality, grow only sub-linearly with time.

Finally, we utilize our proposed algorithms to perform a specific case study of exploiting INT to detect queue occupancy anomalies of network switches using real-world data under realistic settings. We use Mininet [26] to compose networks, and use INT-Path [2] to collect INT data from the one-week time-varying flows [27] deployed in such networks. We use the queue length of each switch as our INT data, and use federated learning implemented by Tensorflow Keras [28] and Federated [29] to train the Long Short-Term Memory (LSTM) models for our detections. Besides, we also use the MNIST [30] data to train a Support Vector Machine (SVM) for handwriting recognization. We observe the following results. Our approach reduces the real-time cost by 34% on average and reduces the long-term cumulative total cost by $15\%\sim57\%$ compared to other algorithms. As a typical machine learning example, our approach can produce LSTM and SVM models that work effectively. Our algorithms can finish execution in seconds.

II. System Model and Problem Formulation

A. Experiment-Based Motivating Study

INT enables switches to insert self-status information into packets that are forwarded via these switches, and existing literatures have already demonstrated that such state information

TABLE II: Major Notations for Model

Inputs	Descriptions[1]
τ	Number of time slots in an interval
$\boldsymbol{\Phi}$	Set of candidate time slots for model aggregations
c_0, c_1, c_2	Unit costs for telemetry, switch, and operation
w_t	Transference cost for global model aggregation at t
$v_{ft},\ r$	Flow rate of f in t, ratio of telemetry rate over flow rate
p	Processing capability per single-unit resource
e_{fi}	Indicator of whether flow f passes switch i
d_{ft}	Difference of telemetry data from f between t and $t-1$
$\theta,\ \varepsilon$	Thresholds of triggering training and model convergence
Decisions	**Descriptions**
x_{ft}	Whether to activate INT for flow f in t
y_{ft}	Amount of resources devoted to flow f in t
z_t	Whether to conduct the global aggregation at t

1. Unless specified otherwise, $t \in \mathcal{T}, f \in \mathcal{F}, i \in \mathcal{N}$ by default.

from network switches can be used for a variety of purposes. In this paper, we will further show that, if we continuously collect a series of INT information from network switches, we can actually conduct machine learning to train models upon such time series INT data and use such trained models to detect anomalies in network switches' queue occupancy. In this section, we demonstrate this new application via experiments.

In our INT implementation, as in Fig. 2, we prepare 160 bits for the INT data for each switch, and such a capacity can be reconfigured if needed. Here, we focus on the length of the packet forwarding queue in each switch as our INT data, where "dep_timedelta" indicates the duration of the packet in the packet queue (in millionseconds). Further information regarding the instantaneous queue length, the congestion status based on the ratio of the current queue length over the configured maximum queue limit, and the number of the dropped packets are all available [31] by extracting the INT data. To specify a route for a packet, we have implemented source routing, in which every 16 bits are used for an egress port and all ports specify the route. The switches forward the packets using the ports pre-stored in the packet header.

We use the flow rate traces from [27], lasting for 7 days and ranging in 0∼12 Gbps, shown in Fig. 3. The left subfigure shows the fluctuations of the (normalized) flow rates; the subfigure in the middle shows the (normalized) flow rates with randomly-inserted anomalous values; the right subfigure shows the (normalized) queue lengths measured by INT at the switches through which the flows with anomalous rates pass.

Learning upon Telemetry: In order to detect the queue length anomalies, we train the Long Short-Term Memory (LSTM) model which is suitable for predicting sequential data over time. We train LSTM via Tensorflow Keras [28], which has an input layer, an output layer, and three hidden layers of 64, 256, and 100 neurons, respectively. The sequential data length of a sample is 3. That is, the LSTM model uses three consecutive data points to predict the follow-up fourth data point in the time series, and performs this in a sliding-window manner. For the results, the squared errors between predicted values and the actual values regarding the queue length are shown in the left subfigure of Fig. 4. If the squared error of one data point is large, it is likely to be an anomaly. We consider it as an anomaly if it exceeds a threshold [32]. The threshold used is 0.03. Here, 96% anomalies are detected, showing that machine learning is effective upon the INT data. For training, we divide the data into mini-batches, where we put 50 data samples as one mini-batch. We pass the entire training data five times, and each time we do it batch by batch for calculating the gradient and updating the model until all samples are handled.

Learning across Sites: The above simple example motivates us to use the INT data for machine learning (e.g., network anomaly detections, or other complex learning-based applications of service providers). However, when services are deployed across multiple geo-distributed serving sites (e.g., Google [33] and Facebook [34]), collecting telemetry data via wide area networks [7] to a central location and conducting machine learning there could be prevented by data locality policies and privacy concerns [8–10], as shown in the right subfigure of Fig. 4. This motivates us to study convergence-preserving *distributed* learning for this case. Our previous Fig. 1 is an indicative visualization of the scenario targeted in this paper. In this figure, the wide-area network connects a cloud with three geo-distributed "sites": the first site on the top left, the second site on the top right, and the third site at the bottom. The third site is further enlarged, and composes the major part of Fig. 1. Each site has its own internal network where INT can be conducted. As shown in the third site, INT information (marked red) is inserted into packets of an existing application flow (marked blue); at the destination of this flow, INT information is separated and used on a dedicated virtual machine instance (marked green) to train the local models (marked in orange). That is, we consider Federated Learning (FL) here: each site is an FL client where the "local training" is conducted, and the cloud is the FL server where the "global aggregation" is conducted. The local models are thus sent to the cloud and aggregated there to produce the global model (marked purple). The global model is then sent back to each site for the local training in the next round.

In this paper, we thus model and design online algorithms to solve the optimization problem of minimizing the cumulative INT overhead and the resource cost of local model training in each site and the traffic footprint of global model aggregations across sites as in federated learning. That is, we control federated learning upon INT data in a cost-minimal manner. We can use this approach to train machine learning models to detect anomalies of network switches' queue occupancy in each site, as elaborated in Section V. More generally, we believe that such INT data can be used to train models to solve other complex problems as well, such as network congestion pattern recognition and crowd traffic prediction.

B. System Settings and Models

We summarize the major notations in Table 2.

In-Band Network Telemetry: We consider multiple distributed locations where each location has a serving site [35] or a point of presence [36] near the users at the network edge, as shown in Fig. 1. These sites are connected via the wide-area network (WAN), and each of such sites has its sophisticated intra-site network (e.g., Fat Tree [37]) which connects a set of switches. Inside each site, a set of flows are deployed, each of which may belong to a different application and travel along only a subset of the switches. We use $\mathcal{N}$ to denote the set of all the switches in all sites, and use $\mathcal{F}$ to denote the set of all

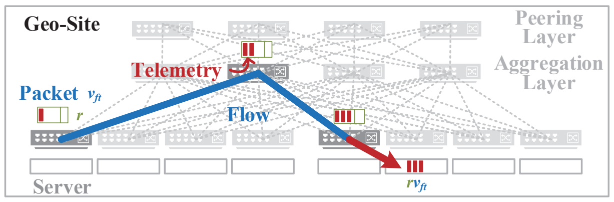


Fig. 5: Modeling for In-band Network Telemetry

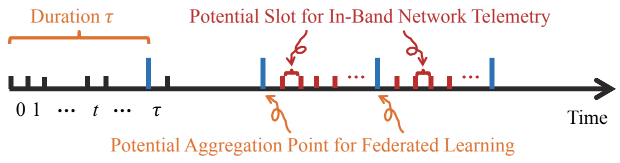

Duration τ
Potential Slot for In-Band Network Telemetry
0 1 ··· t ··· τ
Time
Potential Aggregation Point for Federated Learning

Fig. 6: Control Decisions at Different Frequencies

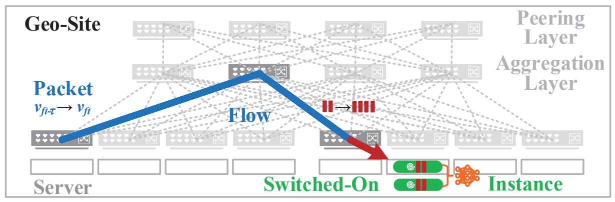


Fig. 7: Additional VM Instance Switched on for Telemetry

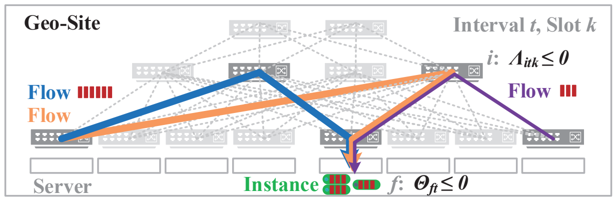


Fig. 8: Switches Covered, and Computation of Telemetry

the flows in all sites. Then, we use the binary indicator e_{fi} to denote whether the flow $f \in \mathcal{F}$ passes the switch $i \in \mathcal{N}$ or not. Enabled by In-Band Network Telemetry (INT) [38], each flow is enabled to collect and host the INT data of a switch in the packet header as the packet passes the switch, whose cost is c_0 per packet. Here, we study the system over a series of time slots $\mathcal{T} = \{0, 1, ..., T\}$. If the rate of the flow f at the time slot t (or the number of packets transported in the flow f during the time slot t) is v_{ft}, then to ensure sufficient measurements, the number of packets used to carry INT data is $v_{ft}r$, where r is a pre-specified ratio, as in Fig. 5.

As an example, we extract related INT data (i.e., queue length [5, 15]) from our testbed by conducting INT upon Mininet [26] with 20 switches, and visualize the results as a bitmap. Visualization results are shown in the experiments later. Other results about the same testbed experiment have been shown in Table 1 previously. Except for visualizations [4, 5], in this paper, we train machine learning models using the INT data via convergence-preserving federated learning.

Federated Learning: We consider multiple sites participating in federated learning. We model federated learning as follows, targeting the FedAvg [39], which is one of the most popular approaches. Each site trains a "local model" using the INT data collected from its own intra-site networks and sends such a local model to a selected central location for the aggregation with local models from other sites, producing a "global model" which is then sent back to every site to continue with the local training in the next round. We use τ to denote the number of consecutive time slots between two adjacent potential global aggregations, as shown in Fig. 6, and use $\mathbf{\Phi} = \{0, \tau, 2\tau, 3\tau...\}$ to denote the set of all the time slots for conducting global aggregations. We also call the time slots of $[t, t+\tau)$ an "interval", $\forall t \in \mathbf{\Phi}$. We denote by c_2 the unit computational cost for processing INT data and training local models, and use w_t to denote the communication cost such as the traffic incurred for global model aggregations at t. We also note that changing the amount of resources can incur the "switching cost" such as the leading time or any system performance oscillation and degradation incurred for instantiating new virtual machine (VM) instances. We use c_1 to denote such cost for switching on one single unit of resources.

Control Decisions: We make three types of decisions. We use $x_{ft} \in \{1, 0\}$ to indicate whether or not to conduct INT for the flow f at the time slot t. We use $y_{ft} \in \mathbb{Z}_{\geq 0}$ to denote the amount of resource such as the number of active virtual machines devoted to federated learning for the flow f at the time slot t. We use $z_t \in \{1, 0\}$ to indicate whether to conduct the global model aggregation across sites at t.

System Cost: The total cost of joint telemetry and federated learning consists of multiple components. Based on the aforementioned notations, first of all, the total overhead of INT data transference is $\sum_{t \in \mathcal{T}} x_{ft} c_0 v_{ft} r \triangleq \sum_{t \in \mathcal{T}} x_{ft} a_{ft}$. Next, the total operational cost of INT data processing and local model training is $\sum_{t \in \mathbf{\Phi}} c_2 y_{ft}$. Further, the total switching cost between adjacent resource provisioning is $\sum_{t \in \mathbf{\Phi}} c_1 [y_{ft} - y_{ft-\tau}]^+$, where $[\cdot]^+ \triangleq \max\{\cdot, 0\}$. As in Fig. 7, due to flow variations, the volume of the extracted telemetry data changes. Accordingly, more computation resources (e.g., additional VM instances) may need to be switched on for training the models. Finally, the total traffic cost of aggregations is $\sum_{t \in \mathbf{\Phi}} z_t w_t$.

Note that the term "cost" in our work does not necessarily refer to monetary cost; instead, we actually use this term to refer to performance cost (or overhead), no matter such cost can be monetized or not. Therefore, the delay to spin up new VMs is a specific example of switching cost, because such delay or lead time impacts service availability and system performance oscillation or degradation. Note that different types of performance costs or overhead can be indeed accumulated, even though they have different units. The right way to accumulate them is not to sum them up by their mathematical values, but to sum them up in a weighted-sum manner—this is an important approach which is pretty common in "multi-objective optimization" and lots of computer networking research literatures. That is, each type of cost (i.e., objective) is associated with or multiplied by a non-negative weight, which is used to indicate the importance and mask the unit difference of this objective, and then multiple weighted objectives are accumulated to compose the overall objective for optimization (i.e., converting multi-objective into single-objective). These weights are also part of inputs (i.e., not decision variables), often pre-specified by the system operator or manager, or the users of the algorithms, based on their own understanding of the problem space and own needs for the optimization.

System Constraints: We enforce multiple constraints. First, for the data collected from INT between two possible consecutive global aggregations (i.e., $\sum_{k\in[t,t+\tau)} x_{fk}v_{fk}r$, $\forall t\in\mathbf{\Phi}$), adequate resources need to be provisioned. Given the processing capability p per unit resource or per single virtual machine, we need to ensure the following inequality:

$$\forall t\in\mathbf{\Phi},\forall f,\Theta_{ft}(\boldsymbol{x}_t,y_{ft})\triangleq(\textstyle\sum_{k\in[t,t+\tau)}x_{fk}v_{fk}r)-y_{ft}p\leq 0.$$

Second, for a network-wise INT, we need to select flows to carry telemetry data to cover every switch of every site at every time slot, where a switch is "covered" if at least one flow with telemetry passes the switch, as shown in Fig. 8:

$$\forall k\in[t,t+\tau),\forall t\in\mathbf{\Phi},\forall i,\ \Lambda_{itk}(\boldsymbol{x}_t)\triangleq 1-\textstyle\sum_f x_{fk}e_{fi}\leq 0.$$

For simplicity, we write its aggregation as $\mathbf{\Lambda}_{it}(\boldsymbol{x}_t)$, where $\boldsymbol{x}_t$ is the aggregation (column vector) of x_{fk}, $\forall k\in[t,t+\tau)$.

Third, since both computation and transmission of federated learning are resource-consuming, we expect to conduct federated learning only when there is sufficient difference between consecutive telemetry results. We use d_{ft} to denote the amount of difference that exist in the INT data collected from the flow f between the time slot t and the time slot $t-1$, and use θ to denote a pre-specified threshold. If the difference between the INT data from the current time slot and that from the previous time slot is larger than the threshold, then we should continue to do INT at the next time slot; otherwise, we do not have to do INT. To capture this, as shown in Fig. 9, we enforce

$$\forall k-1,k\in[t,t+\tau),\forall t\in\mathbf{\Phi},\forall f,||x_{fk-1}d_{fk-1}-\theta||\leq(Mx_{fk}+\xi),$$

where M is a large constant and $\xi<\theta$ is another small constant. Besides, at any time slot t, if INT is conducted, then a follow-up global aggregation needs to be conducted at a later time slot as soon as possible. To capture this,

$$\forall k-1,k\in[t,t+\tau),\forall t,t+\tau\in\mathbf{\Phi},\forall f,x_{fk}-z_{t+\tau}\leq 0.$$

Combing the above two, we define $\Delta_{ftk}(x_{fk-1},x_{fk},z_{t+\tau})$:

$$[||x_{fk-1}d_{fk-1}-\theta||-(Mx_{fk}+\xi);x_{fk}-z_{t+\tau}]^\top.$$

For the ease of the presentation, we express the constraints as $\mathbf{\Delta}_{\boldsymbol{ft}}(\boldsymbol{x}_t,z_t)\preceq\mathbf{0}$, $\forall t$, where $\mathbf{\Delta}_{\boldsymbol{ft}}$ is the aggregation of Δ_{ftk} and is convex in $\boldsymbol{x}_t$ (convex) and z_t (linear).

Fourth, in order to guarantee the convergence of the global model being trained by federated learning across sites, the number of global model aggregations needs to be no less than a desired threshold (i.e., $\sum_{t\in\mathbf{\Phi}}z_t\geq\mathcal{O}(\varepsilon)$). That is, the global accuracy ε is achieved after $\mathcal{O}(\varepsilon)$ aggregations [20, 40]. Here, $\mathcal{O}$ refers to asymptotical growth. Convergence is a natural requirement in most, if not all, of the cases of (supervised) machine learning. The nature of (supervised) machine learning is minimizing the underlying loss function, and thus one would desire that the algorithm used to minimize the loss function can indeed eventually minimize it to its theoretical optimum, i.e., to converge. More importantly, one would desire to relate the extent to which the loss function is minimized with the amount of computation needed by the algorithm. In federated learning, the amount of computation could refer to the number of global aggregations that aggregate the local models and the number of local iterations (e.g., gradient decent) that computes each local model (note that in this work, we do not control the number of local iterations per time slot, and assume it is given; that is, with a given number of local iterations per time slot, the convergence can be related with the number of global aggregations alone). Overall, convergence ensures the "quality" of the model to be trained.

C. Problem Formulation and Challenges

Control Problem $\mathbb{P}$: With the above models, we formulate the following optimization problem of minimizing the system's total cost of INT with convergence-preserving federated learning in a long-term time scope:

$$\min \sum_{t\in\boldsymbol{\mathcal{T}};f}x_{ft}a_{ft}+\sum_{t\in\mathbf{\Phi}}\Big\{\sum_f\{c_1[y_{ft}-y_{ft-\tau}]^+ + c_2y_{ft}\}+z_tw_t\Big\}$$

$$\begin{aligned}
s.t.\ & \Theta_{ft}(\boldsymbol{x}_t,y_{ft})\leq 0, && \forall t\in\mathbf{\Phi},\forall f\in\boldsymbol{\mathcal{F}}, && (1)\\
& \mathbf{\Lambda}_{\boldsymbol{it}}(\boldsymbol{x}_t)\preceq\mathbf{0}, && \forall t\in\mathbf{\Phi},\forall i\in\boldsymbol{\mathcal{N}}, && (2)\\
& \mathbf{\Delta}_{\boldsymbol{ft}}(\boldsymbol{x}_t,z_t)\preceq\mathbf{0}, && \forall t\in\mathbf{\Phi},\forall f\in\boldsymbol{\mathcal{F}}, && (3)\\
& \textstyle\sum_{t\in\mathbf{\Phi}}z_t\geq\mathcal{O}(\varepsilon), && && (4)\\
var.\ & x_{ft}\in\{0,1\},y_{ft}\in\mathbb{Z}_{\geq 0},z_t\in\{0,1\}, && && (5)
\end{aligned}$$

whose objective function is denoted as $\mathcal{P}$. We have explained Constraints (1)∼(4) sequentially in the above. Constraint (5) specifies the domains of our control variables.

We actually maintain a weight for each term in the optimization objective as these terms in the objective represent different types of costs. We do not show these weights in the problem formulation for the ease of the presentation.

The problem $\mathbb{P}$ is "useful" yet "challenging". It is useful because collecting INT data and training machine learning models can be exploited for lots of purposes, including network diagnosis (e.g., anomaly detection) as we will demonstrate later; conducting such machine learning across sites can make the model more general and accurate by training it over more diversified, larger volumes of INT data. This is however a challenging and non-trivial problem because INT data collection in each site and distributed model training (as in federated learning) incurs overhead continuously, and such overhead needs to be carefully managed or optimized over time. Specifically, when making the control decisions online at each time slot regarding whether to conduct INT for each network flow, how much computation resources to allocate for INT processing and local model training, and whether to conduct the global aggregation (as required in federated learning), we face the challenges of decision coupling in adjacent time slots, long-term constraints over time, and the problem's NP-hardness, further elaborated as follows.

Algorithmic Challenges: We actually face multiple challenges when designing algorithms to solve problem $\mathbb{P}$ online.

First, in the online setting, we need to make the control decisions irrevocably in real time as the system runs, and can only make such decisions for each time slot based on the inputs for that time slot without any inputs beyond that time slot. This constitutes an obstacle. For example, at the time slot $t-\tau$, in order to minimize $[y_{ft}-y_{ft-\tau}]^+$ as in $\mathcal{P}$, we need to set a value to $y_{ft-\tau}$; however, as the value of y_{ft} is unknown

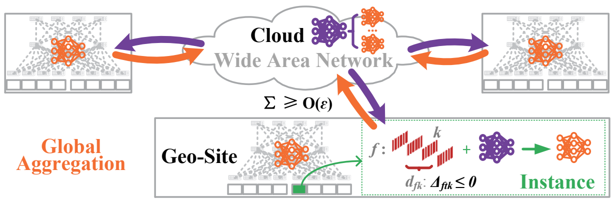

Cloud
Wide Area Network
$\Sigma \geqslant O(\varepsilon)$
Global Aggregation
Geo-Site
f:
k
d_{fk}: $\Delta_{fk} \leq 0$
Instance

Fig. 9: Global Model Aggregation across Multiple Sites

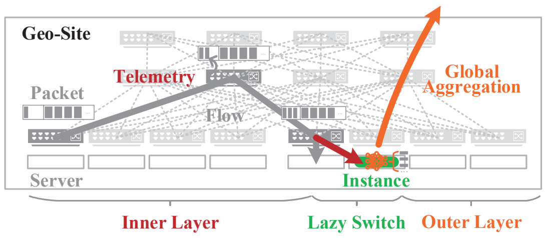


Fig. 10: Provisioning for Different Frequencies

(it is a future decision that can only be made at t), it is hard to choose a good value for $y_{ft-\tau}$ at $t-\tau$.

Second, Constraints (1) and (4) are in the "long-term" form which imposes a non-trivial challenge as well. In (1), having made the decision of y_{ft} at t, we need to determine x_{fk} at each $k \in [t, t+\tau)$ to ensure the sum of $x_{fk}v_{fk}r$ across $[t, t+\tau)$ is no greater than $y_{ft}p$. This is in fact a dilemma, as we have no idea on how v_{fk} varies over k: choosing a larger x_{fk} (i.e., $x_{fk}=1$) at the beginning will force a smaller x_{fk} (i.e., $x_{fk}=0$) in the future, which cannot be optimal if v_{fk} decreases over k; choosing a smaller x_{fk} at the beginning can force a larger x_{fk} in the future due to Constraint (2), which cannot be optimal as well if v_{fk} increases over k. It is an analogous case for (4), where we need to guarantee that the cumulative sum of z_t is no less than a given constant for ensured model convergence.

Third, this problem $\mathbb{P}$ is a non-linear integer program, which is actually NP-hard, via reducing the "covering" problem to $\mathbb{P}$ regarding those terms with $\{x_{fk}\}$. This intractability is already not easy to handle in the offline setting where we are assumed to see all the inputs altogether before the system starts. Solving this problem in an online manner without observing all the inputs of the entire time horizon increases the difficulty.

Algorithmic Goal: We aim to design the polynomial-time online approximation algorithms for the problem $\mathbb{P}$ to find the solutions $\{\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}}\}$ while provably ensuring $\mathcal{P}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}}) \leq \gamma \mathcal{P}^*$, where $\mathcal{P}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}})$ is the objective function value evaluated with our online solutions; $\mathcal{P}^*$ is the offline optimal objective function value, assuming all the inputs are observed in prior; and γ is a constant called the "competitive ratio", which measures the gap against the optimum.

Note that the goal of our work is neither the fine-tuning of the hyper-parameters of any machine learning models, nor the selection or determination regarding what could be the best model for any learning tasks; instead, we optimize the cost of training, regardless of what specific machine learning models are trained in the federated learning framework.

III. Algorithm Design

Preliminaries: We introduce auxiliary notations to facilitate our algorithm design, and then present the algorithm overview.

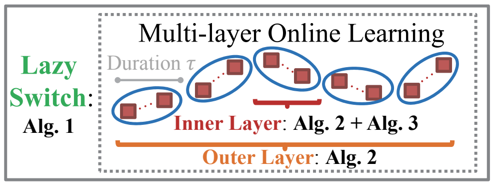


Fig. 11: Lazy Switch with Online Learning

First, we split the objective function. For $t \in \boldsymbol{\Phi}$, we define

$$C^t_{\neg S}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) \triangleq \textstyle\sum_{k\in[t,t+\tau);f} x_{fk}a_{fk} + \sum_f c_2 y_{ft} + z_t w_t,$$
$$C^t_S(\boldsymbol{y}_t, \boldsymbol{y}_{t-\tau}) \triangleq \textstyle\sum_f c_1[y_{ft} - y_{ft-\tau}]^+,$$

which represent the non-switching cost and the switching cost, respectively. For the constraints, we define

$$\boldsymbol{g}^s_{t,1}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) = \boldsymbol{\Lambda_{it}}(\boldsymbol{x}_t),$$
$$\boldsymbol{g}^l_{t,1}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) = \tau\mathcal{O}(\varepsilon)/T - z_t,\ \boldsymbol{\Delta_{ft}}(\boldsymbol{x}_t, z_t)$$

corresponding to the short-term (i.e., for every time slot) constraint and the long-term constraint, respectively, for the large time scale of all $t \in \boldsymbol{\Phi}$. Similarly, we then define the following, for all $k \in [t, t+\tau)$ of a given $t \in \boldsymbol{\Phi}$, with short-term and long-term constraints:

$$\boldsymbol{g}^s_{k,2}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) = \Lambda_{ik}(\boldsymbol{x}_t),$$
$$\boldsymbol{g}^l_{k,2}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) = x_{fk}v_{ft}r - y_{ft}p/\tau,\ \Delta_{ftk}(x_{fk-1}, x_{fk}, z_{t+\tau}),$$

Next, we formulate the following subproblem, where we add the new constraint (6) and relax $\boldsymbol{x}_t$ and z_t:

$$\begin{aligned}
\min\ & \textstyle\sum_{t\in\boldsymbol{\Phi}} \mathcal{P}_{t,1} \triangleq \sum_{t\in\boldsymbol{\Phi}} C^t_{\neg S}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) \\
s.t.\ & C^t_S(\boldsymbol{y}_t, \boldsymbol{y}_{t-\tau}) \leq \zeta_1 C^t_{\neg S}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t), \quad \forall t \in \boldsymbol{\Phi}, \qquad (6) \\
& \boldsymbol{g}^s_{t,1}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) \preceq \boldsymbol{0}, \quad \forall t \in \boldsymbol{\Phi}, \\
& \textstyle\sum_{t\in\boldsymbol{\Phi}} \boldsymbol{g}^l_{t,1}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) \preceq \boldsymbol{0}. \\
var.\ & x_{ft} \in [0,1], y_{ft} \in \mathbb{R}_{\geq 0}, z_t \in [0,1],
\end{aligned}$$

We also introduce the following subproblem for each interval. That is, $\forall t \in \boldsymbol{\Phi}$, given $\boldsymbol{y}_t$ and z_t, we have

$$\begin{aligned}
\min\ & \textstyle\sum_{k\in[t,t+\tau)} \mathcal{P}_{k,2} \triangleq C^t_{\neg S}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) \\
s.t.\ & \boldsymbol{g}^s_{k,2}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) \preceq \boldsymbol{0},\ \forall k \in [t, t+\tau), \\
& \textstyle\sum_{k\in[t,t+\tau)} \boldsymbol{g}^l_{k,2}(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) \preceq \boldsymbol{0}, \\
var.\ & x_{ft} \in [0,1],
\end{aligned}$$

where $\mathcal{P}_{k,2} := x_{fk}a_{fk} + (\sum_f c_2 y_{ft} + z_t w_t)/\tau$, $\forall k \in [t, t+\tau)$.

Although the problems have long-term objectives, the one-shot versions are $\mathbb{P}_{t,1} := \min \mathcal{P}_{t,1}$, $\mathbb{P}_{k,2} := \min \mathcal{P}_{k,2}$.

Algorithm Overview: We design Algorithm 1 as our major algorithm, which invokes Algorithms 2 and 3, as in Fig. 10 and 11. Algorithm 1 controls resources via the "lazy switch", and uses Algorithm 2 for both inner and outer "online learning". Algorithm 3 rounds fractional decisions to integers.

A. Lazy Switch with Multi-Layer Online Learning

Algorithm 1: Algorithm 1 postpones the switch operation that changes the amount of the resources (e.g., the number of virtual machines) until the cumulative non-switching cost so

Algorithm 1 Lazy Switch with Online Learning

```
1: Initialize $t = t' = 0$; given $\bar{\boldsymbol{y}}_{-\tau} = \bar{\boldsymbol{y}}_0 = \boldsymbol{0}$;
2: while $t \in \boldsymbol{\Phi}$ do
3:   if $C_S^{t'}(\bar{\boldsymbol{y}}_{t'}, \bar{\boldsymbol{y}}_{t'-\tau}) \le \frac{1}{\zeta_0} \sum_{v \in \boldsymbol{\Phi} \cap [t', t)} C_{\neg S}^{v}(\bar{\boldsymbol{y}}_v, \bar{\boldsymbol{x}}_v, \bar{z}_v)$ then
4:     Obtain $\widetilde{\boldsymbol{y}}_t$, $\widetilde{\boldsymbol{x}}_t$, $\widetilde{z}_{t+\tau}$ from $\mathbb{P}_{t,1}$ via Algorithm 2;
5:     Round $\widetilde{\boldsymbol{y}}_t$ to obtain $\bar{\boldsymbol{y}}_t$;
6:     If $\bar{\boldsymbol{y}}_t \ne \bar{\boldsymbol{y}}_{t-\tau}$, $t' = t$;
7:   end if
8:   for $k \in [t, t+\tau)$ do
9:     Obtain $\widetilde{\boldsymbol{x}}_k$ from $\mathbb{P}_{k,2}(\cdot, \bar{\boldsymbol{y}}_t, \widetilde{z}_{t+\tau})$ via Algorithm 2;
10:    Round $\widetilde{\boldsymbol{x}}_k$ to get $\bar{\boldsymbol{x}}_k$ via Algorithm 3;
11:  end for
12:  $\bar{z}_{t+\tau} = 1$ if $\boldsymbol{\Delta_{ft}}(\boldsymbol{x}_t, \cdot)$ requires else round $\widetilde{\boldsymbol{z}}_{t+\tau}$;
13: end while
```

far, including the telemetry overhead, the resource operational cost, and the global model aggregation cost, exceeds a specified control parameter ζ_0 times the switching cost of the most recent switch operation. This comparison of switching cost vs. non-switching cost is in Line 3. When switching is needed, our algorithm seeks the potentially new control decisions in Line 4. These new decisions are subject to Constraint (6), so that the switching cost, if the switching operation is to occur currently, will not exceed another specified control parameter ζ_1 times the non-switching cost for the current time slot. Via ζ_0 and ζ_1 that control the "laziness" of the switch operations, Algorithm 1 proceeds "conservatively" while overcoming the blindness to the varying inputs of the future in the online setting.

For each $t \in \boldsymbol{\Phi}$, Algorithm 1 obtains the control decisions from the subproblem $\mathbb{P}_{t,1}$. Note that we still need to satisfy the long-term constraint $\sum_{t \in \boldsymbol{\Phi}} \boldsymbol{g}_{t,1}^l(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) \preceq \boldsymbol{0}$, which is hard to do in the online setting as discussed in "Algorithmic Challenges" previously. Therefore, instead of seeking to satisfy it strictly, we allow it to be violated; but we will upper-bound the cumulative violation with provable guarantees and more importantly, will ensure that the time-averaged violation will diminish as the length of the entire time horizon goes to infinity. This is "online learning" as we will discuss, and we call this per-interval online learning as the "outer" online learning. The approach of lazy switch actually coordinates the inner and the outer online learning via controlling the VM instances for distributed learning.

For each interval $k \in [t, t+\tau), \forall t \in \boldsymbol{\Phi}$, proposed Algorithm 1 obtains the control decisions from the subproblem $\mathbb{P}_{k,2}$ in Line 9, where we face related long-term constraint $\sum_{k \in [t,t+\tau)} \boldsymbol{g}_{k,2}^l(\boldsymbol{x}_t, \boldsymbol{y}_t, z_t) \preceq \boldsymbol{0}$. Analogously, we apply the online learning idea, and we call this per-time-slot online learning as the "inner" online learning. Although $\{\boldsymbol{\Delta_{ft}}\}$ is treated as long-term constraint, it is actually ensured by the algorithm after being solved as shown in the analysis.

B. Online Learning under Long-Term Constraints

Algorithm 2: Algorithm 2 solves both $\mathbb{P}_{t,1}$ and $\mathbb{P}_{k,2}$ on the fly by using a common framework of online learning. Note that a convex optimization with the objective function $\sum_v o_v(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}})$, the long-term constraint $\sum_v \boldsymbol{g}_v^l(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}) \preceq \boldsymbol{0}$, and the short-term constraints $\boldsymbol{g}_v^s(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}) \preceq \boldsymbol{0}, \forall v$ (we call it "short-term", as it is for every slot v individually) can be equivalently re-written into the following convex-concave format:

$$\min_{\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}} \max_{\boldsymbol{\lambda}_v} \sum_v \left(o_v(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}) + \boldsymbol{\lambda}_v^\top \boldsymbol{g}_v^l(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}})\right), s.t., \boldsymbol{g}_v^s(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}) \preceq \boldsymbol{0}.$$

Here, all the decision variables are in the real domain, and $\{\boldsymbol{\lambda}_v \in \mathbb{R}_{\ge 0}^{dim(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}})}\}$ are the Lagrange multipliers. In the online setting, we can only solve this problem by using the observable inputs at each v as time goes, where v represents the time slot. Towards that end, we consider the following function by introducing the Lagrange multiplier:

$$\mathcal{L}_v(\widetilde{\boldsymbol{\mathcal{I}}}, \boldsymbol{\lambda}) = o_v(\widetilde{\boldsymbol{\mathcal{I}}}) + \boldsymbol{\lambda}^\top \boldsymbol{g}_v^l(\widetilde{\boldsymbol{\mathcal{I}}}).$$

That is, we alternately minimize $\mathcal{L}_v(\widetilde{\boldsymbol{\mathcal{I}}}, \boldsymbol{\lambda}_{v+1})$ with respect to $\widetilde{\boldsymbol{\mathcal{I}}}$ with the given $\boldsymbol{\lambda}_{v+1}$ and maximize $\mathcal{L}_v(\widetilde{\boldsymbol{\mathcal{I}}}_v, \boldsymbol{\lambda})$ with respect to $\boldsymbol{\lambda}$ with the given $\widetilde{\boldsymbol{\mathcal{I}}}_v$. We can then achieve such an alternate optimization through primal descent steps and dual ascent steps iteratively over time. As the primal step, we solve the following subproblem:

$$\begin{aligned} \min_{\boldsymbol{\mathcal{I}}} \quad & \nabla o_v(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}})^\top (\boldsymbol{\mathcal{I}} - \widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}) + \boldsymbol{\lambda}_{v+1}^\top \boldsymbol{g}_v^l(\boldsymbol{\mathcal{I}}) + \tfrac{||\boldsymbol{\mathcal{I}} - \widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}||^2}{2\alpha}, \\ s.t., \quad & \boldsymbol{g}_{v+1}^s(\boldsymbol{\mathcal{I}}) \preceq \boldsymbol{0}, \end{aligned} \tag{7}$$

where α is a positive step size. The objective function here is just an approximation to $\mathcal{L}_v(\widetilde{\boldsymbol{\mathcal{I}}}, \boldsymbol{\lambda}_{v+1})$. As the dual step, we update the Lagrange multiplier as follows:

$$\boldsymbol{\lambda}_{v+1} = [\boldsymbol{\lambda}_v + \mu \nabla \boldsymbol{g}_v^l(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}})]^+, \tag{8}$$

where the step size μ used here is also a positive constant.

We highlight two aspects. First, the objective function of the problem (7) is not a standard gradient-based approximation to $\mathcal{L}_v(\widetilde{\boldsymbol{\mathcal{I}}}, \boldsymbol{\lambda}_{v+1})$, but rather a rectified approximation by introducing a carefully-designed "regularization" term $\frac{||\boldsymbol{\mathcal{I}} - \widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}||^2}{2\alpha}$, in order to facilitate our performance analysis as demonstrated later. Second, the constraint of the problem (7) is changing as time goes, while traditional online learning approaches often only support the unchanged domains. Despite the step

Algorithm 2 Online Learning

```
// For $\mathbb{P}_{t,1}$, $\boldsymbol{\mathcal{I}_t} = (\boldsymbol{x}_t^\top, \boldsymbol{y}_t^\top, z_t)^\top$, $o_t = \mathcal{P}_{t,1}$, $\boldsymbol{g}_t^s = \boldsymbol{g}_{t,1}^s$, $\boldsymbol{g}_t^l = \boldsymbol{g}_{t,1}^l$;
// For $\mathbb{P}_{k,2}$, $\boldsymbol{\mathcal{I}_k} = \boldsymbol{x}_k$, $o_k = \mathcal{P}_{k,2}$, $\boldsymbol{g}_k^s = \boldsymbol{g}_{k,2}^s$, $\boldsymbol{g}_k^l = \boldsymbol{g}_{k,2}^l$;
1: Initialize $\boldsymbol{\lambda}_0 = \boldsymbol{0}$, and set proper step sizes to $\alpha$ and $\mu$;
2: for $v$ do
3:   Observe $\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}$, $o_v$, and $\boldsymbol{g}_v^l$;
4:   Update $\boldsymbol{\lambda}_{v+1}$ by (8), and then solve $\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v+1}}$ from (7);
5: end for
```

Algorithm 3 Randomized Rounding, $\forall t \in \boldsymbol{\Phi}$

```
Input: $\forall i, \sum_f \widetilde{x}_{ft} e_{fi}$ as an integer;
1: Determined: $\Upsilon \leftarrow \{\}$; Undetermined: $\Gamma_i \leftarrow \{\}, \forall i$;
2: for $i \in \boldsymbol{\mathcal{N}}$ do
3:   $\Gamma_i \triangleq \{f \mid e_{fi} = 1\} \backslash \Upsilon$; $\Sigma_i \triangleq \sum_f \widetilde{x}_{ft} e_{fi} - \sum_{f \in \Upsilon} \bar{x}_{ft} e_{fi}$;
4:   while True do
5:     $\{\bar{x}_{ft} | f \in \Gamma_i\} \leftarrow$ DepRound($\Sigma_i$, $\{\widetilde{x}_{ft} | f \in \Gamma_i\}$);
6:     if $\forall i', |\Gamma_{i'} \backslash \Gamma_i| \ge \Sigma_{i'} - \sum_{f \in \Gamma_{i'} \cap \Gamma_i} \bar{x}_{ft}$ then
         Update $\Upsilon = \Upsilon \cup \Gamma_i$, $\Gamma_i \leftarrow \{\}$, and break;
7:   end while
8: end for
```

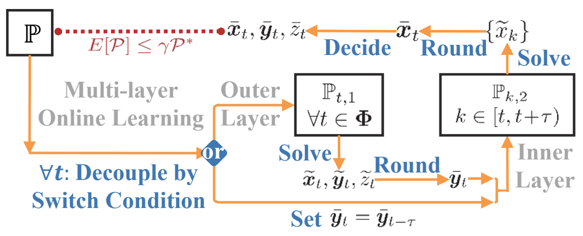

$\mathbb{P}$
$E[\mathcal{P}] \leq \gamma\mathcal{P}^*$
$\bar{\boldsymbol{x}}_t, \bar{\boldsymbol{y}}_t, \bar{z}_t$
Decide
$\bar{\boldsymbol{x}}_t$
Round
$\{\widetilde{x}_k\}$
Solve
Multi-layer Online Learning
Outer Layer
$\mathbb{P}_{t,1}$ $\forall t \in \boldsymbol{\Phi}$
$\mathbb{P}_{k,2}$ $k \in [t, t+\tau)$
or
Solve
Round
Inner Layer
$\forall t$: Decouple by Switch Condition
$\widetilde{\boldsymbol{x}}_t, \widetilde{\boldsymbol{y}}_t, \widetilde{z}_t$
$\bar{\boldsymbol{y}}_t$
Set $\bar{\boldsymbol{y}}_t = \bar{\boldsymbol{y}}_{t-\tau}$

Fig. 12: Roadmap of Proposed Algorithms and Results

sizes set by following previous works [41, 42], we extend the theoretical analysis to time-varying domains for our problem.

We clarify again the notations of $\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}$, o_v and $\boldsymbol{g}_v^l$ appearing in Line 3 of Algorithm 2 ($v = t$ for $\mathbb{P}_{t,1}$ and $v = k$ for $\mathbb{P}_{k,2}$). Specifically, for the problem $\mathbb{P}_{t,1}$, they mean the following:

$$\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{t}} = (\widetilde{\boldsymbol{x}}_t^\top, \widetilde{\boldsymbol{y}}_t^\top, \widetilde{z}_t)^\top, \quad o_t = \mathcal{P}_{t,1}(\widetilde{\boldsymbol{x}}_t, \widetilde{\boldsymbol{y}}_t, \widetilde{z}_t),$$
$$\boldsymbol{g}_t^l = (\tau\mathcal{O}(\varepsilon)/T - \widetilde{z}_t, \ (\boldsymbol{\Delta_{ft}}(\widetilde{\boldsymbol{x}}_t, \widetilde{z}_t))^\top)^\top.$$

That is, in the round t (as in Line 2 of Algorithm 2) where we will solve the solution for $t+1$, we need to observe (i) the existing solution that has already been produced by Algorithm 2 for t, (ii) the objective function $\mathcal{P}_{t,1}$, and (iii) the corresponding constraint functions. These inputs are required for constructing the auxiliary problem (7) as stated in Line 4 of Algorithm 2. Analogously, for the problem $\mathbb{P}_{k,2}$, these notations mean the following:

$$\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{k}} = \widetilde{\boldsymbol{x}}_k, \quad o_k = \mathcal{P}_{k,2}(\widetilde{\boldsymbol{x}}_k, \widetilde{\boldsymbol{y}}_t, \widetilde{z}_t),$$
$$\boldsymbol{g}_k^l = (x_{fk}v_{ft}r - \widetilde{y}_{ft}p/\tau, \Delta_{ftk}(x_{fk-1}, x_{fk}, \widetilde{z}_{t+\tau}))^\top,$$

with analogous interpretations as the above. Note that for $k \in [t, t+\tau)$, $\widetilde{\boldsymbol{y}}_t$ and $\widetilde{z}_t$ are given (as already determined at t).

C. Randomized Rounding with Preservation

Algorithm 3: Algorithm 3 rounds fractional decisions into integers, which is invoked in Line 10 of Algorithm 1. Then, Algorithm 3 invokes a randomized dependent rounding algorithm in its Line 5, DepRound [43], as follows:

$$(v_1, v_2) \to \begin{cases} (v_1 + \varpi_1, v_2 - \varpi_1), \ with\ probability\ \frac{\varpi_2}{\varpi_1 + \varpi_2} \\ (v_1 - \varpi_2, v_2 + \varpi_2), \ with\ probability\ \frac{\varpi_1}{\varpi_1 + \varpi_2} \end{cases},$$

where $\varpi_1 = \min\{1 - v_1, v_2\}$ and $\varpi_2 = \min\{v_1, 1 - v_2\}$. Given the sum Σ^i of some fractions $\{\widetilde{x}_{ft} | f \in \Upsilon_i\}$, DepRound rounds each fraction into an integer of either 0 or 1 such that i) the sum of the integers after rounding is still Σ^i, and ii) the expectation of every integer is actually the corresponding fraction before rounding (i.e., $E[\bar{\boldsymbol{x}}_t] = \widetilde{\boldsymbol{x}}_t$).

To ensure $\sum_f \widetilde{x}_{ft} e_{fi}, \forall i$ are all integers, we may need to adapt the constant on the right-hand side of the constraint $\sum_f x_{ft} e_{fi} \geq 1$. That is, "1" is used for all i at first, and then for some i, the constant is increased by 1 and $\widetilde{\boldsymbol{x}}_t$ needs to be re-solved. As the maximum number of flows is limited and each increment is at least one, the number of such increments is also limited. We should mention here that the $\widetilde{\boldsymbol{x}}_t$ that makes $\sum_f \widetilde{x}_{ft} e_{fi}$ integral always exists upon such increasement.

In Algorithm 1 Line 5, $\widetilde{\boldsymbol{y}}_t$ is rounded "conventionally", by rounding it up to 1 with the probability of $\widetilde{y}_{ft}$ and rounding it down to 0 with the probability of $1 - \widetilde{y}_{ft}$. In Algorithm 1 Line 12, if $\boldsymbol{\Delta_{ft}}$ requires $\bar{z}_t$ to be 1, then we set $\bar{z}_t = 1$; otherwise, $\bar{z}_t$ could set to either 0 or 1, where we will use conventional rounding again to ensure $E[\bar{z}_t] = \widetilde{z}_t$, and also $E[\bar{z}_t] \geq \widetilde{z}_t$.

IV. Performance Analysis

Preliminaries: We need some notations to present our analysis. $\{\bar{\boldsymbol{x}}_t, \bar{\boldsymbol{y}}_t, \bar{z}_t\}$ is the (integral) output of our proposed online approach. $\{\widetilde{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}_t, \widetilde{z}_t\}$ is the (fractional) output of multi-layer online learning. $\widetilde{\boldsymbol{x}}_t$ solved from $\mathbb{P}_{t,1}$ is updated later by solving $\mathbb{P}_{k,2}$. Given $\bar{\boldsymbol{y}}_t$ and $\widetilde{z}_t$, $\{\widetilde{\boldsymbol{x}}_t^\diamond\}$ is the optimal solution to $\sum_{k \in [t, t+\tau)} \mathcal{P}_{k,2}$. Given $\bar{\boldsymbol{y}}_t$, $\{\widetilde{\boldsymbol{x}}_t^*, \widetilde{z}_t^*\}$ is the optimal solution to $\sum_{t \in \boldsymbol{\Phi}} \mathcal{P}_{t,1}$. $\{\boldsymbol{x}^*, \boldsymbol{y}^*, z^*\}$ is offline optimum.

Before presenting our theoretical results, we need to make the following assumptions to facilitate the analysis:

Assumption 1: All domains used are bounded, and all of the objectives (if derivable) have bounded gradients. The objective and constraints for online learning are all convex.

Assumption 2: The optimum exists per slot and interval. The changes in the long-term constraints are bounded.

Analysis Overview: We can interpret our theoretical results as follows. For the problem, Theorems 2 and 4, and Lemma 1 are for objectives; Theorems 1, 3, and 5 are for constraints. For the algorithms, Theorem 1 is for inner online learning; Theorems 2 and 3 are for outer online learning; Lemma 1, and Theorems 4 and 5 are for lazy switch with multi-layer online learning, plus rounding. Theorem 4 is our major conclusion.

Theorem 1. *Given $\bar{\boldsymbol{y}}_t, \forall t \in \boldsymbol{\Phi}$, the violation of the long-term constraint (1) for $[t, t+\tau)$ grows sub-linearly ($\gamma'_{t,0} < 1$):*

$$\forall f, [E[\Theta_{ft}(\bar{\boldsymbol{x}}_t, \bar{y}_{ft})]]^+ \leq [\Theta_{ft}(\widetilde{\boldsymbol{x}}_t, \bar{y}_{ft})]^+ \leq \mathcal{O}(\tau^{\gamma'_{t,0}}).$$

Proof. See Section 4.1. Via inner online learning. □

Theorem 2. *Given $\bar{\boldsymbol{y}}_t, \forall t \in \boldsymbol{\Phi}$, the regret of non-switching cost over the entire time horizon $\mathcal{T}$ grows sub-linearly ($\gamma_1 < 1$):*

$$\sum_{t \in \boldsymbol{\Phi}} \{E[C_{\neg S}^t(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}_t^\diamond, \widetilde{z}_t)] - C_{\neg S}^t(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}_t^*, \widetilde{z}_t^*)\} \leq \mathcal{O}(T^{\gamma_1}).$$

Proof. See Section 4.2. Via outer online learning. □

Theorem 3. *The violation of the long-term constraint (4) for $\mathcal{T}$ (i.e., the model converges) grows sub-linearly ($\gamma'_1 < 1$):*

$$E[\sum_{t \in \boldsymbol{\Phi}} \bar{z}_t] \geq \sum_{t \in \boldsymbol{\Phi}} \widetilde{z}_t \geq \mathcal{O}(\varepsilon) - \mathcal{O}(T^{\gamma'_1}).$$

Proof. See Section 4.3. Via outer online learning. □

Lemma 1. *The cumulative non-switching cost and the cumulative switching cost satisfies the following inequality:*

$$\sum_{t \in \boldsymbol{\Phi}} C_S^t(\bar{\boldsymbol{y}}_t, \bar{\boldsymbol{y}}_{t-\tau}) \leq \gamma'_2 \sum_{t \in \boldsymbol{\Phi}} C_{\neg S}^t(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}_t^*, \widetilde{z}_t^*),$$

where γ'_2 is a constant. By letting $\gamma_2 = \gamma'_2(1 + \max\{\zeta_1, 1/\zeta_0\})$, we have $\mathcal{P}(\widetilde{\boldsymbol{x}}^*, \bar{\boldsymbol{y}}, \widetilde{z}^*) \leq \gamma_2 \mathcal{P}(\boldsymbol{x}^*, \boldsymbol{y}^*, z^*) = \gamma_2 \mathcal{P}^*$.

Proof. See Section 4.4. Via the lazy switch approach. □

Theorem 4. *The competitive ratio γ of our proposed approach is shown through the following inequality:*

$$E[\mathcal{P}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}})] \leq \{\gamma_2 \mathcal{P}^* + \mathcal{O}(\Psi_\tau + (T/\tau)^{\gamma_1})\} \triangleq \gamma\mathcal{P}^*,$$

where Ψ_τ, γ_1 and γ_2 are all constants, given T and τ.

Proof. See Section 4.5. Via Theorem 2 and Lemma 1. □

Theorem 5. *$\{\bar{\boldsymbol{x}}_t, \bar{\boldsymbol{y}}_t, \bar{z}_t\}$ satisfy the constraints (2) and (3).*

Proof. See Section 4.6. Via expectation preservation. □

A. Proof of Theorem 1

Proof. We show the result by first introducing a proposition:

Proposition 1. *For the optimization with objective $\sum_v o_v(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}})$ and long-term constraint $\sum_v \boldsymbol{g}_v(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}) \preceq \mathbf{0}$, the main results obtained by using online learning are shown as follows:*

$$\textstyle\sum_v \{o_v(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}) - o_v(\widetilde{\boldsymbol{\mathcal{I}}}^*_{\boldsymbol{v}})\} \le \mathcal{O}(V^{\alpha_1}), ||[\sum_v \boldsymbol{g}_v(\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}})]^+|| \le \mathcal{O}(V^{\alpha_2}),$$

where the variable $\widetilde{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}$ is the aggregation of all decisions over a convex domain $\widetilde{\boldsymbol{\mathcal{X}}} \subseteq \mathbb{R}^{|\widetilde{\mathcal{I}}|}$; the overall scope is V; $\alpha_1 < 1$, $\alpha_2 < 1$ are constants; and $\{\widetilde{\boldsymbol{\mathcal{I}}}^*_{\boldsymbol{v}}\}$ are the dynamic optimums.

The details regarding such proposition for online learning are omitted, since they have been studied by the literatures before upon some simple assumptions (convex & bounded) [41, 42, 44–46] just mentioned before.

We summary necessary assumptions as follows:

Assumption 1: o_v has bounded gradient, i.e., $||\nabla o_v(\widetilde{\boldsymbol{\mathcal{I}}})|| \le F, \forall \widetilde{\boldsymbol{\mathcal{I}}} \in \widetilde{\mathcal{X}}$; and $\mathbf{g}_v(\widetilde{\boldsymbol{\mathcal{I}}})$ is bounded, i.e., $||\mathbf{g}_v(\tilde{\mathbf{I}})|| \le G$.

Assumption 2: There exists a constant $\varepsilon' > 0$, and an interior point $\widehat{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}} \in \widetilde{\mathcal{X}}$ such that $\forall v, \mathbf{g}_v(\widehat{\boldsymbol{\mathcal{I}}}_{\boldsymbol{v}}) \le -\varepsilon' \mathbf{1}$.

Assumption 3: Slack constant ε' satisfies: $\varepsilon' > \overline{V}(\mathbf{g})$, where the point-wise maximal variation of consecutive constraints is $\overline{V}(\mathbf{g}) := \max_v \max_{\widetilde{\boldsymbol{\mathcal{I}}} \in \widetilde{\mathcal{X}}} ||[\mathbf{g}_{v+1}(\widetilde{\boldsymbol{\mathcal{I}}}) - \mathbf{g}_v(\widetilde{\boldsymbol{\mathcal{I}}})]^+||$.

Assumption 1 bounds primal and dual gradients, which is a very common assumption [47]. Assumption 2 is Slater's condition, which guarantees the existence of a bounded optimal Lagrange multiplier. Assumption 3 implies that the slack constant is larger than the maximal variation of the constraints, requiring $\min_{u,v} \max_{\widetilde{\boldsymbol{\mathcal{I}}} \in \widetilde{\mathcal{X}}} [-g_{u,v}(\widetilde{\boldsymbol{\mathcal{I}}})]^+$ is larger than $\overline{V}(\mathbf{g})$, when feasible region defined by $\mathbf{g}_v(\widetilde{\boldsymbol{\mathcal{I}}}) \preceq \mathbf{0}$ is large enough, or the trajectory of $\mathbf{g}_v(\widetilde{\boldsymbol{\mathcal{I}}})$ is smooth enough.

Furthermore, given $\bar{\boldsymbol{y}}_t$, $\forall f, ||[\Theta_{ft}(\widetilde{\boldsymbol{x}}_t, \bar{y}_{ft})]^+|| \le \mathcal{O}(\tau^{\gamma'_{t,0}})$ is obtained from **Proposition 1**, where $\gamma'_{t,0}$ is a constant and $\gamma'_{t,0} < 1$. We then have $[E[\Theta_{ft}(\bar{\boldsymbol{x}}_t, \bar{y}_{ft})]]^+ = [\Theta_{ft}(\widetilde{\boldsymbol{x}}_t, \bar{y}_{ft})]^+$, since Θ_{ft} is a linear function respect to $\{x_{ft}\}$. □

B. Proof of Theorem 2

Proof. For $\mathbb{P}_{t,1}$, let $\widetilde{\boldsymbol{\mathcal{X}}}_t$ be the domain just defined by Constraint (6) and $\boldsymbol{g}^s_{t,1}$. Note that $\bar{\boldsymbol{x}}_t$ is solved under $\widetilde{\boldsymbol{\mathcal{X}}}_t$ and $\{\widetilde{\boldsymbol{x}}^\diamond_t\}$ are the dynamic optimums. Then, for a minimum objective, we have $E[C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}^\diamond_t, \widetilde{z}_t)] \le E[C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}_t, \widetilde{z}_t)]$, where $\bar{\boldsymbol{y}}_t$ is given and $\widetilde{z}_t$ is also solved from $\mathbb{P}_{t,1}$. Then, the left terms in Theorem 2 is actually less than the following term $\sum_{t\in\boldsymbol{\Phi}} \{E[C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}_t, \widetilde{z}_t)] - C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}^*_t, \widetilde{z}^*_t)\}$.

Via **Proposition 1** for a series of $\mathbb{P}_{t,1}$ over $t \in \boldsymbol{\Phi}$, we have

$$\textstyle\sum_t \{E[C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}_t, \widetilde{z}_t)] - C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}^*_t, \widetilde{z}^*_t)\} \le$$
$$\textstyle\sum_t \{E[C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}_t, \widetilde{z}_t)] - C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}^{**}_t, \widetilde{z}^{**}_t)\} \le \mathcal{O}((\tfrac{T}{\tau})^{\gamma_1}),$$

where $\gamma_1 < 1$ is a constant; $\widetilde{\boldsymbol{x}}^*_t$ and $\widetilde{z}^*_t$ are both the dynamic optimums over $\widetilde{\boldsymbol{\mathcal{X}}}_t$; $\widetilde{\boldsymbol{x}}^{**}_t$ and $\widetilde{z}^{**}_t$ are the dynamic optimums over $\widetilde{\boldsymbol{\mathcal{X}}} = \cup_{t\in\boldsymbol{\Phi}} \widetilde{\boldsymbol{\mathcal{X}}}_t$. The first inequality sign holds due to the optimum over a larger domain. The second inequality holds due to the online learning over $|\boldsymbol{\Phi}| = T/\tau$ subproblems. □

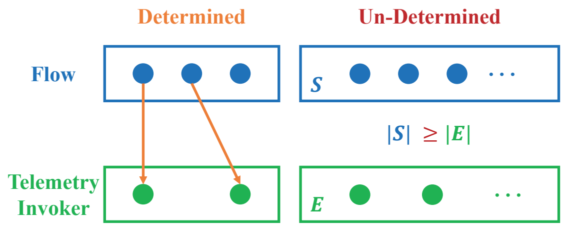


Fig. 13: Loop Invariant for Adopting DepRound

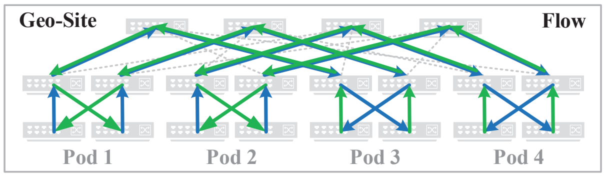


Fig. 14: Topology and Inter-Pod Flows

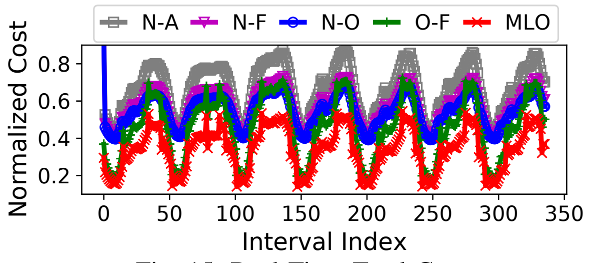


Fig. 15: Real-Time Total Cost

C. Proof of Theorem 3

Proof. When deciding $\bar{z}_t$, there are two basic cases. i) $\bar{z}_t$ is 1 when a large difference exists on consecutive telemetry data. Then, $\bar{z}_t = 1 \ge \widetilde{z}_t$. ii) $\bar{z}_t$ could be either 0 or 1. Then, $\bar{z}_t$ is rounded upon $\widetilde{z}_t$ (i.e., $E[\bar{z}_t] = \widetilde{z}_t$). By combining these two cases, we have $E[\bar{z}_t] \ge \widetilde{z}_t, \forall t \in \boldsymbol{\Phi}$. After that, we sum up this inequality over T and have $\sum_{t\in\boldsymbol{\Phi}} E[\bar{z}_t] \ge \sum_{t\in\boldsymbol{\Phi}} \widetilde{z}_t$.

Since $\boldsymbol{g}^l_{t,1} = \tau\mathcal{O}(\varepsilon)/T - \widetilde{z}_t$ is also the long-term constraint of $\mathbb{P}_{t,1}$ and $\widetilde{z}_t$ is solved, by using **Proposition 1** again, we then have $||[\sum_{t\in\boldsymbol{\Phi}} \boldsymbol{g}^l_{t,1}]^+|| = ||[\mathcal{O}(\varepsilon) - \sum_{t\in\boldsymbol{\Phi}} \widetilde{z}_t]^+|| \le \mathcal{O}(T^{\gamma'_1})$, where γ'_1 is a constant and $\gamma'_1 < 1$. There are also two cases here. In the first case, $\mathcal{O}(\varepsilon) - \sum_{t\in\boldsymbol{\Phi}} \widetilde{z}_t \le 0$. Therefore, we have $\sum_{t\in\boldsymbol{\Phi}} \widetilde{z}_t \ge \mathcal{O}(\varepsilon)$, which directly obeys the Corollary needed to be proved. In the second case, $\mathcal{O}(\varepsilon) - \sum_{t\in\boldsymbol{\Phi}} \widetilde{z}_t > 0$. Then, we have $||\mathcal{O}(\varepsilon) - \sum_{t\in\boldsymbol{\Phi}} \widetilde{z}_t|| \le \mathcal{O}(T^{\gamma'_1})$. Although $\sum_{t\in\boldsymbol{\Phi}} \widetilde{z}_t$ is less than $\mathcal{O}(\varepsilon)$ in this case, the distance between them is not large (i.e., we have the following result $\sum_{t\in\boldsymbol{\Phi}} \widetilde{z}_t \ge \mathcal{O}(\varepsilon) - \mathcal{O}(T^{\gamma'_1})$), which ensures the number of global aggregations.

$\mathcal{O}(\varepsilon)$ is larger than $\mathcal{O}(T^{\gamma'_1})$ since the federated learning often contains hundreds of global model aggregations. $\mathcal{O}(T^{\gamma'_1})$ is actually $\mathcal{O}((T/\tau)^{\gamma'_1})$. And $\boldsymbol{g}^l_{t,1}$ could be scaled for a desired value, since $\sum_{t\in\boldsymbol{\Phi}} \boldsymbol{g}^l_{t,1}/M \le 0$ is equivalent to $\sum_{t\in\boldsymbol{\Phi}} \boldsymbol{g}^l_{t,1} \le 0$ for optimization. M is a large number. □

D. Proof of Lemma 1

Proof. Given $\bar{\boldsymbol{y}}_t$ and $\widetilde{z}_t, \forall t \in \boldsymbol{\Phi}$, constraint $\boldsymbol{g}^s_{k,2}(\boldsymbol{x}_t, \bar{\boldsymbol{y}}_t, \widetilde{z}_t)$ decides a series of domains $\widetilde{\boldsymbol{\mathcal{X}}}_k \subseteq \widetilde{\boldsymbol{\mathcal{X}}}, \forall k \in [t, t+\tau)$. Then

$$\textstyle\sum_{k\in[t,t+\tau)} \mathcal{P}_{k,2}(\widetilde{\boldsymbol{x}}^\diamond_t) \ge \sum_{k\in[t,t+\tau)} \mathcal{P}_{k,2}(\widetilde{\boldsymbol{x}}^{\diamond *}_t),$$

where $\widetilde{\boldsymbol{x}}_t^{\diamond *}$ is dynamic optimums over $\widetilde{\boldsymbol{\mathcal{X}}}$, and the previous inequality holds since $\widetilde{\boldsymbol{\mathcal{X}}}$ is a larger domain. We have

$$\textstyle\sum_k\{\mathcal{P}_{k,2}(\widetilde{\boldsymbol{x}}_t)-\mathcal{P}_{k,2}(\widetilde{\boldsymbol{x}}_t^{\diamond})\} \leq \sum_k\{\mathcal{P}_{k,2}(\widetilde{\boldsymbol{x}}_t)-\mathcal{P}_{k,2}(\widetilde{\boldsymbol{x}}_t^{\diamond *})\}.$$

$\{\widetilde{\boldsymbol{x}}_t\}$ is obtained by a series of $\mathbb{P}_{k,2}$ over $k \in [t, t+\tau)$ via online learning. Then, by using **Proposition 1**, the right part of the previous inequality is bounded by $\mathcal{O}(\tau^{\gamma_{t,0}})$, where $\gamma_{t,0} < 1$ is a constant. Then, given $\bar{\boldsymbol{y}}_t$, we consider the difference for $\mathcal{P}_{k,2}(\widetilde{\boldsymbol{x}}_t, \widetilde{z}_t)$ and $\mathcal{P}_{k,2}(\widetilde{\boldsymbol{x}}_t, \bar{z}_t)$ (i.e., $(\bar{z}_t - \widetilde{z}_t)w_t$). Since the value of $\bar{z}_t$ is either 0 or 1, $(\bar{z}_t - \widetilde{z}_t)w_t \leq (1-\widetilde{z}_t)w_t$. Since $\widetilde{z}_t$ is solved by the outer online learning, $(1-\widetilde{z}_t)w_t$ is a constant Ξ. For $E[C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \bar{\boldsymbol{x}}_t, \bar{z}_t)] - C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}_t^{\diamond}, \widetilde{z}_t)$, it is $\leq \mathcal{O}(\tau^{\gamma_{t,0}})$, where $E[\bar{\boldsymbol{x}}_t] = \widetilde{\boldsymbol{x}}_t$ is ensured by rounding.

Here, we introduce some new notations:

$$\begin{aligned}(C^v_S)_1 = C^v_S(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}^*_t, \widetilde{z}^*_t),\ \ (C^v_{\neg S})_1 = C^v_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}^*_t, \widetilde{z}^*_t),\\ (C^v_S)_2 = C^v_S(\boldsymbol{y}^*_t, \boldsymbol{x}^*_t, z^*_t),\ \ (C^v_{\neg S})_2 = C^v_{\neg S}(\boldsymbol{y}^*_t, \boldsymbol{x}^*_t, z^*_t).\end{aligned}$$

For the switching cost incurred between two consecutive switches (i.e., $\forall t_u$ and $\forall t_{u+1} \in \boldsymbol{\Phi}$, $\forall u : 1 \leq u \leq u'$), the non-switching cost is at least ζ_0 times the switching cost, as shown in the algorithm, where u' is the maximum index recorded for switching. Further, the potential switching cost in $[t_{u'}, t]$ is at most ζ_1 times the non-switching cost. That is, $\forall t \in \boldsymbol{\Phi}$,

$$\begin{aligned}&\textstyle\sum_{v=0}^t (C^v_S)_1 = \sum_{u\leq u'}\sum_{v=t_u}^{t_{u+1}-1}(C^v_S)_1 + \sum_{v=t_{u'}}^t (C^v_S)_1\\ &\textstyle\leq \sum_{u\leq u'}\left\{\frac{1}{\zeta_0}\sum_{v=t_u}^{t_{u+1}-1}(C^v_{\neg S})_1 + 0\right\} + \left\{\zeta_1 (C^{t_{u'}}_{\neg S})_1 + 0\right\}\\ &\textstyle\leq \max\{\zeta_1, 1/\zeta_0\}\sum_{v=0}^t (C^v_{\neg S})_1,\end{aligned}$$

where we omit the related decisions for simplicity. And, after defining $\gamma_2' = \max_{v\in\boldsymbol{\Phi}} \frac{\max C^v_{\neg S}}{\min C^v_{\neg S}}$ over the domain of the integer decisions, we have $C^v_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}^*_t, \widetilde{z}^*_t) \leq \gamma_2' C^v_{\neg S}(\boldsymbol{y}^*_t, \boldsymbol{x}^*_t, z^*_t)$. By summing up the previous inequality over $T, \forall t \in \boldsymbol{\Phi}$, we have the following inequality for the optimums of reals and integers:

$$\textstyle\sum_{v=0}^t C^v_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}^*_t, \widetilde{z}^*_t) \leq \gamma_2' \sum_{v=0}^t C^v_{\neg S}(\boldsymbol{y}^*_t, \boldsymbol{x}^*_t, z^*_t).$$

The overall cost over the entire time horizon is

$$\begin{aligned}&\textstyle\mathcal{P}(\widetilde{\boldsymbol{x}}^*, \bar{\boldsymbol{y}}, \widetilde{z}^*) \leq (1+\max\{\zeta_1, 1/\zeta_0\})\sum_{t\in\boldsymbol{\Phi}} C^t_{\neg S}(\widetilde{\boldsymbol{x}}^*, \bar{\boldsymbol{y}}, \widetilde{z}^*),\\ &\textstyle\leq \gamma_2'(1+\max\{\zeta_1, 1/\zeta_0\})\sum_{t\in\boldsymbol{\Phi}}\{(C^t_{\neg S})_2 + (C^t_S)_2\} \triangleq \gamma_2 \mathcal{P}^*,\end{aligned}$$

where $\gamma_2 = \gamma_2'(1+\max\{\zeta_1, 1/\zeta_0\})$ forms the conclusion. □

E. Proof of Theorem 4

Proof. We complete this theorem by introducing the following chain of derivations, which combines previous results

$$\begin{aligned}&\textstyle E[\mathcal{P}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}})] = E[\sum_{t\in\boldsymbol{\Phi}}\{C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \bar{\boldsymbol{x}}_t, \bar{z}_t) + C^t_S(\bar{\boldsymbol{y}}_t, \bar{\boldsymbol{y}}_{t-\tau})\}]\\ &\textstyle\leq E[\sum_{t\in\boldsymbol{\Phi}}\{C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}^{\diamond}_t, \widetilde{z}_t) + \mathcal{O}(\tau^{\gamma_{t,0}}) + C^t_S(\bar{\boldsymbol{y}}_t, \bar{\boldsymbol{y}}_{t-\tau})\}]\\ &\textstyle\leq \sum_{t\in\boldsymbol{\Phi}}\{C^t_{\neg S}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{x}}^*_t, \widetilde{z}^*_t) + C^t_S(\bar{\boldsymbol{y}}_t, \bar{\boldsymbol{y}}_{t-\tau}) + \mathcal{O}(\tau^{\gamma_{t,0}})\} + \mathcal{O}(T^{\gamma_1})\\ &\leq \{\gamma_2\mathcal{P}^* + \mathcal{O}\big(\Psi_\tau + (T/\tau)^{\gamma_1}\big)\} \triangleq \gamma\mathcal{P}^*,\end{aligned}$$

where the first inequality sign holds because of Lemma 1; the second inequality holds by applying Theorem 2; and the third inequality holds based on the results also from Lemma 1. We define $\Psi_\tau = \sum_{t\in\boldsymbol{\Phi}}\mathcal{O}(\tau^{\gamma_{t,0}})$. We should mentioned here that Ψ_τ and γ are both constants, given T and τ. □

TABLE III: Environments and Data Traces

Environment/Trace	Description
Mininet [26]	4 Sites, Each with 20 Switches
Flow [27]	Flow Rates 0∼12 Gbps, 7 Days
Telemetry [2]	INT Packets over Flow Rate as 20
Instance [50]	0.093\$/hour for AWS EC2 "m3.medium"
Learning [12, 28]	LSTM for Flow, SVM for MNIST [30]

F. Proof of Theorem 5

Proof. $\{\boldsymbol{\Delta_{ft}}\}$ is directly ensured by proposed algorithm after being solved. For constraint $\{\boldsymbol{\Lambda_{it}}\}$, line 6 of Algorithm 3 enures that, those undetermined variables always form a feasible solution of DepRound. That is, $\forall i \in \mathcal{N}$, the sum of those undetermined variables is not larger than their amount. As in Fig. 13, the number of undetermined flows $|S|$ should be larger than the number of undetermined telemetry invoker $|E|$, in case flows are sufficient for telemetry decision. DepRound can let some of the variables be 1 to ensure the unchanged sum and $E[\bar{\boldsymbol{x}}_t] = \widetilde{\boldsymbol{x}}_t$. Such condition is similar to the Hall Theorem [48] for determining whether a matching exists in a bipartite graph [49] via repeatedly checking the numbers of the vertexes and the number of their neighbors. □

V. Experimental Evaluations

A. Evaluation Settings

We summarize the environment and the data in Table 3.

Sites, Flows, and INT: We use Mininet [26] to construct 4 sites, each with 20 switches connected via a fat tree [37]. Each fat tree has 4 pods and 4 aggregation switches, where each pod has 4 top-of-rack switches. We obtain the flow trace from [27], with the flow rates of 0∼12 Gbps, where the trace is recorded every quarter (i.e., 15 minutes) through 7 days. We use this trace to assign 8 flows to each site, with 4 flows traveling from the first two pods to the last and 4 other flows in the opposite direction, as shown in Fig. 14. We adopt the implementation of "INT-Path" [2] for INT. We focus on the queue length of each switch's packet queue as our INT data. Aligned with INT-Path, where 50 packets are used to carry INT data for a flow of 1000 packets per second, we set the ratio of INT packets over the flow rate as 20 [2]. The INT packet has a field "int_lists", where each element in this list corresponds to a switch. The data collected from a switch includs "ingress_port", "egress_port", "swid", "deq_timedelta", "deq_qdepth" and time stamps, where "deq_timedelta" records the time waiting in the queue and "deq_qdepth" records the queue length of the switch.

Federated Learning: We conduct the federated learning upon the INT data for queue occupancy anomaly detection. That is, given the queue length time series data of a switch, we train a Long Short-Term Memory (LSTM) model to detect the existence and the number of anomalies in the queue length variations. To create the ground truth, we choose one flow and add noise to its flow rate to create queue length anomalies; then, we extract the INT data of queue length from all flows for training and detection. We randomly choose 30 out of the total 672 quarters to add such noise, where the maximum noise value matches the maximum flow rate in our trace. We train the LSTM model via Tensorflow Keras [28] and Tensorflow

Federated [29]. And the settings for the LSTM are the same as in Section II-A (i.e., mentioned in our preliminary case study). We highlight that the anomaly detection using INT data is just a case study here; further exploitations of INT data via federated learning can be indeed studied in the future.

Besides anomaly detection for the switch queue occupancy, we additionally use the MNIST [30] data, which consist of 70k images of handwritten digits, to train a least-square Support Vector Machine (SVM) model [12] via our federated learning to detect whether the single digit in an image is even or odd. Note that the visualization derived from INT is similar to the images in the MNIST dataset, and we use MNIST for digit recognition to simulate the learning task of using INT for network congestion pattern detection, as discussed later.

More specifically, the loss of the SVM (with the norm for smoothing) is $\{\lambda * pow(linalg.norm(w), 2) + val/len\}/2$, where λ is set to 0.1 as the learning rate; $linalg$ is a function of the Python library "Numpy"; w is a vector indicating the parameter of the SVM; len is the size of the samples; and val is the sum of $pow(max(0, 1-y_i * np.inner(w, x_i)), 2)$. Here, $np.inner$ calculates the inner product; x_i is the feature of the sample i; and y_i is the label of the sample i. For LSTM, there are three hidden layers. We use the "Dropout" method with the parameter 0.2 as regularization. The activation mode after the output layer is linear, and the model is compiled with the "rmsprop" optimizer. We use "model.fit" for training, whose inputs are the features and labels of the samples, the epoch size, the batch size, and the "validation_split" value of 0.5.

Resource Cost: The operational cost for a virtual machine is set to 0.093\$/hour, which is the price of an AWS EC2 virtual machine "m3.medium" with 1 vCPU and 3.75-GB memory [50]. The unit switching cost is set as doubling the operational cost. Since INT is on a per minute basis and a virtual machine can often last for tens of minutes, we specify that a virtual machine lasts for two quarters (i.e., 1 minute is one time slot and 30 time slots constitute an interval). Regarding the weights associated to the different cost terms in the objective, we set these weights such that the weighted costs fall into the same order of magnitude in terms of their mathematical values.

Algorithms: We denote our proposed approach that uses lazy switch and multi-layer online learning (via AMPL [51] and IPOPT [52] to solve the subproblems) as MLO in the evaluation results. We define a "mod" file containing the problem formulation via AMPL. Then, we use the Python package "amplpy" to import the "mod" files (one file per subproblem), set the solver as IPOPT, and use the DataFrame API of AMPL to combine the model formulation with the input data that dynamically arrive at runtime. After repetitively calling "ampl.solve()", we obtaine the results in an online manner. We also compare different combinations of alternative algorithms[1]. For determining the number of virtual machines per time slot, the algorithms we use are as follows:

- N-$*$ uses a fixed number of virtual machines for the entire time horizon. We set the default number of virtual machines to 7, and vary it in our evaluations.
- O-$*$ uses lazy switch and outer online learning. We use the parameters of $\zeta_0 = 2$ and ζ_1 = 0.5, derived from [53], as the inputs in our experiments.

For determining whether to conduct INT per time slot or at a different frequency, the algorithms are as follows:

- $*$-A triggers INT in every minute with a given ratio of INT packets to the flow rate. This ratio is 20.
- $*$-F triggers INT at a fixed frequency (i.e., every three minutes), unless data of adjacent time slots show adequate difference where INT is done.
- $*$-O uses inner online learning for INT within an interval, with the number of virtual machines given by the other algorithm described in the above. We currently use the parameters of $\mu = 0.15$ and $\alpha = 2$.

B. Evaluation Results

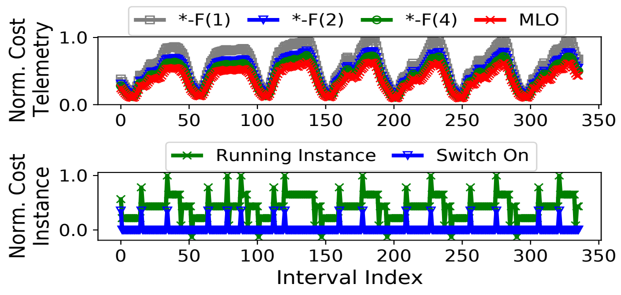
Fig. 15 first shows the total cost per interval. MLO always performs the best, reducing the total cost by 34% on average compared to others. Compared to those without online learning, MLO reduces at least 60% total cost. The fluctuations of the total cost mainly depend on the dynamic flow rates which impact the telemetry overhead and further impact the cost of the resources for the federated learning.

Note that Fig. 15 and Fig. 3 (as shown previously) convey different information. The left subfigure in Fig. 3 shows the fluctuations of the (normalized) flow rates; the subfigure in the middle of Fig. 3 shows the (normalized) flow rates with randomly-inserted anomalous values; the right subfigure in Fig. 3 shows the (normalized) queue lengths measured by telemetry for the switches which the flows with anomalous rates pass through. In contrast, Fig. 15 shows the "system cost" in each time slot incurred by different control algorithms, where the system cost, as defined in Section II-B, includes telemetry overhead; operational cost and switching cost of resources for telemetry data processing and local model training; and traffic of global model aggregations as in federated learning. The goal of this paper is to optimize the cumulative system cost in the long run. In this sense, Fig. 3 shows information about the input, and Fig. 15 exhibits the output. Another difference is that Fig. 3 only shows flow traces in part of the first day, and Fig. 15 shows the system cost over the entire time horizon of 7 days.

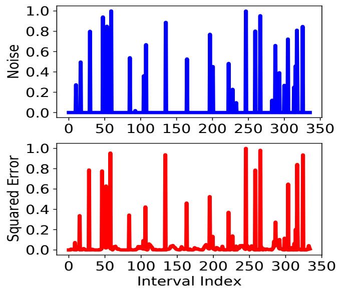
Fig. 16(a) depicts how telemetry and resource costs impact the total cost. MLO always performs the best, decreasing the overall cost by 15.8%$\sim$57.9% compared to other approaches. This figure also illustrates, for MLO, the switching cost using the blue line and the corresponding operational cost with green line. MLO dynamically changes the number of virtual machines and triggers INT to minimize the cost.

Fig. 16(b) shows the training results of federated learning upon INT data. The top figure shows the ground-truth normalized flow rate, where the peaks are our added noises. For queue length, we conduct federated learning to train an LSTM across four sites (with normal flows from other sites), and calculate the squared error in the bottom figure. The peaks in errors imply that we can indeed detect anomalies in queue length (e.g., for a threshold 0.05). The count of true positive

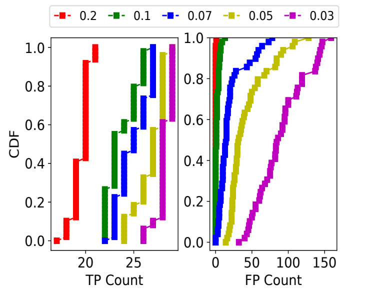
[1]Here, the mark $*$ is a wildcard to match different algorithms. The notation "$*(n)$" shows the parameters (i.e., we run the algorithm with its algorithmic parameter being set as the value n in the experiments).



(a) Impact of Flow Rate

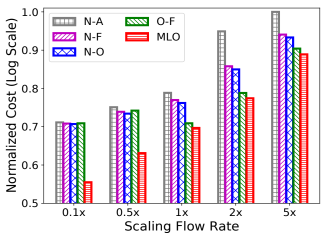


(b) Detection upon Telemetry

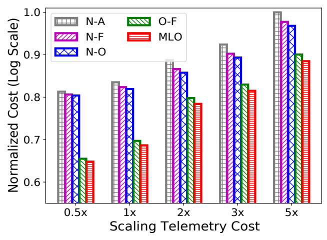


(c) Impact of Learning Threshold

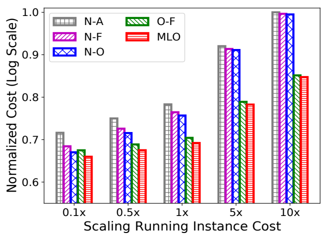
Fig. 16: Results of Our Approach



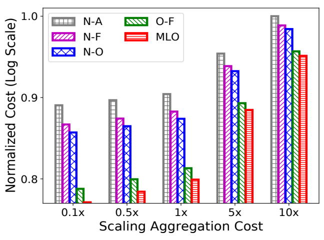
(a) Impact of Flow Rate



(b) Impact of Telemetry Cost



(c) Impact of Operational Cost



(d) Impact of Aggregation Cost

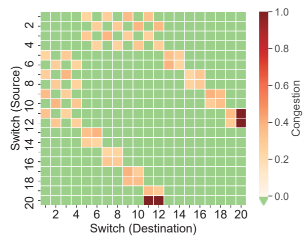
Fig. 17: Cumulative Total Cost of Different Approaches

(TP) is 28 and the count of false positive is 23. The ratio of true positive to all noises is 28/30=93%; and the ratio of false positive to all data points is 23/672=3%.

Fig. 16(c) shows the results regarding both true positive and false positive under various thresholds. When the threshold is large, fewer noises are detected. The ratio of true positive to all noises (average for multiple trials) is 77%, and the ratio of false positive to all data points is less than 3%. To balance the detection and false alarms, one may choose to pre-train the model upon a small proportion of the datasets.

Fig. 17 exhibits the cumulative total cost over the entire time horizon for different approaches. In Fig. 17(a), with the growth of flow rates, the cumulative cost increases. To ensure the ratio of INT packets over the flow's, the growth of the flow rate leads to the growth of the INT cost, which also demands more resources for federated learning. The average cost reduction by MLO compared to others is 68.7%. In Fig. 17(b), with the growth of INT overhead, the cumulative cost increases. Since telemetry overhead is relatively small compared to that of O-F, the average cost reduction of MLO is only 8%, without increasing the flow rates. Note that O-F uses outer online learning for virtual machine resource control and conducts telemetry at a fixed frequency. In Fig. 17(c), with the growth of the operational cost, the cumulative cost increases. Compared to N-O, the cost reduction of MLO is 7%∼1.93x. Note that N-O uses inner online learning and a fixed number of virtual machines. In Fig. 17(d), with the growth of the global model aggregation cost, the cumulative cost increases. The average cost reduction of MLO compared to others is 49.6%.

Fig. 18 visualizes the queue lengths in a two-dimensional image, and a sample image from the MNIST dataset. In the left subfigure, the horizontal and the vertical axes are "Switch IDs". The element at the a-th row and the b-th column of the matrix represents the maximum queue length of the switches "en route" recorded by the INT packets traveling from Switch a to Switch b within a specified time period. We normalize these maximum queue lengths, and consider them as indicators for congestion—the higher the maximum queue length is, the more congested the network path is. This is similar to the MNIST data in the right subfigure. For example, the left subfigure contains the English letter "U" and the right subfigure contains the number "4". This indicates that, similar to digit recognition, using the INT-based two-dimensional images as the features and the specified network congestion patterns (unnecessarily letters or numbers) as the labels, we can adopt our proposed algorithms to train models for detecting network congestion patterns, or more broadly, for the scenarios of network diagnosis and other related cases.



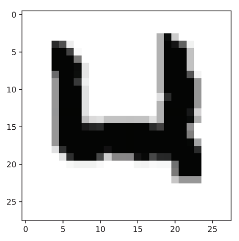
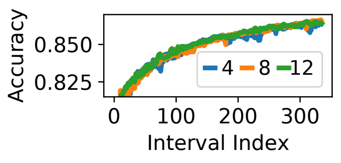


Fig. 18: Visualizations for Telemetry and MNIST



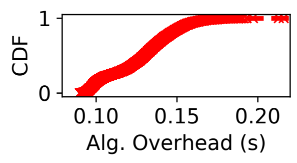


Fig. 19: Results of Our Approach upon MNIST Data

Fig. 19 adopts the MNIST data to train SVMs for digit recognition under MLO to simulate the task of training SVMs for network congestion pattern detection. The learning task here is to distinguish even and odd numbers. We train multiple SVMs across 4, 8, and 12 participants (i.e., sites in our case), respectively, in federated learning and all of the trained models converge. The data samples of MNIST are randomly divided among the sites, as mentioned in the implementation [12]. It also visualizes that the average execution time incurred by our proposed approach is only 0.13s, which is totally acceptable. Note that this is using MNIST for digit recognition to simulate INT for congestion pattern detection, and does not mean in reality we should use MNIST data to replace INT data; in reality, we should still use INT data, and here we choose to use the MNIST data to simulate the INT data because we lack sufficient amounts of INT data for this specific evaluation scenario (although we use real-world flow data for the anomaly detection scenario). Also, the MNIST evaluations train SVMs to demonstrate that our proposed algorithms also work with convex loss functions, in addition to non-convex loss functions.

VI. Related Work

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

A. Optimization for In-Band Network Telemetry

In-Band Network Telemetry (INT) was comprehensively researched in research [3, 54]. INT [38] was a telemetry application using P4, which obtained the device-internal states without disturbing the controller. LightGuardian [1] first conducted a full-visibility and lightweight INT system using a small constant-sized data structure. PINT [14] was designed as an INT framework that bounded the amount of information added to each packet. NetView [15] supported various telemetry applications and telemetry frequencies on demand, monitoring each device via sending dedicated probes [54]. INT-path [2] decoupled the system into a routing mechanism and a routing path generation policy for embedding source routing into INT probes with minimum paths.

These works consider the implementation and the optimization for INT, but fail to consider higher-level applications or complicated machine learning using data from INT.

B. Optimization for Federated Learning

Some works [17, 18, 55] discussed the theoretically analysis for federated learning. Wang $et\ al.$ [12] proposed a control algorithm that determined the best trade-off between local updates and global parameter aggregation under a given resource budget. The network-aware optimization of distributed learning was proposed [19] for the fog computing, where devices optimally shared local data processing and sent their learnt parameters to a server for periodic aggregation. Zhou $et\ al.$ [20] investigated how to coordinate the edge and the cloud to optimize the system-wide cost efficiency of federated learning. Guo $et\ al.$ [21] adjusted the parameters for learning, that minimized the overall cost. Lots of other existing papers focus on the practical design and implementation of federated learning systems [11], producing machine learning models without theoretically characterizing the convergence aspect.

These works either focus on the optimization of federated learning but fail to consider sources and training data collection, especially obtained frequently from a faster time scale (e.g., INT), or focus on system design and implementation, being unable to incorporate the learning convergence.

C. Online Provisioning over Clusters

Hung $et\ al.$ [7] studied related management of multiple resources over wide area network. Zhou $et\ al.$ [22] proposed a learning-driven cloud resource provisioning for the content providers. SmartFCT [24] employed the learning coupled with software defined networking to improve the power efficiency. Zhao $et\ al.$ [25] found that the completion time for flows could be further optimized considering the destinations as an additional dimension via reducer placement. Clarinet [56] conducted the analytics over wide area network and dynamically adopt the best one with the minimum response time. Yuan $et\ al.$ [10] treated the critical communication bottleneck.

These works optimize various applications [6, 57, 58] and conduct the resource management for clusters, and do not explore applications such as federated learning or INT.

VII. Conclusion

Federated learning and INT have been often investigated separately. In this paper, we bridge this gap by studying the scenario of conducting federated learning across sites while provisioning resources and scheduling INT to collect training data in each site. We formulate a non-linear integer program to optimize the long-term total cost of the system, and solve it on the fly by the laziness-based switch control approach with multi-layer online learning and randomized rounding. We prove a diverse set of performance guarantees rigorously, and use trace-driven experiments to validate that our approach can reduce the total cost significantly and can train models to solve real problems with high accuracy.

References

[1] Y. Zhao, K. Yang, Z. Liu, T. Yang, L. Chen, S. Liu, N. Zheng, R. Wang, H. Wu, Y. Wang *et al.*, "Lightguardian: A full-visibility, lightweight, in-band telemetry system using sketchlets." in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2021, pp. 991–1010.

[2] T. Pan, E. Song, Z. Bian, X. Lin, X. Peng, J. Zhang, T. Huang, B. Liu, and Y. Liu, "Int-path: Towards optimal path planning for in-band network-wide telemetry," in *IEEE Conference on Computer Communications (INFOCOM)*, 2019, pp. 487–495.

[3] L. Tan, W. Su, W. Zhang, J. Lv, Z. Zhang, J. Miao, X. Liu, and N. Li, "In-band network telemetry: a survey," *Elsevier Computer Networks (CN)*, vol. 186, pp. 1–28, 2021.

[4] Z. Liu, J. Bi, Y. Zhou, Y. Wang, and Y. Lin, "Netvision: Towards network telemetry as a service," in *IEEE International Conference on Network Protocols (ICNP)*, 2018, pp. 247–248.

[5] C. Guo, L. Yuan, D. Xiang, Y. Dang, R. Huang, D. A. Maltz, Z. Liu, V. Wang, B. Pang, H. Chen, Z. Lin, and V. Kurien, "Pingmesh: A large-scale system for data center network latency measurement and analysis," in *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, 2015, pp. 139–152.

[6] E. Cortez, A. Bonde, A. Muzio, M. Russinovich, M. Fontoura, and R. Bianchini, "Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms," in *ACM Symposium on Operating Systems Principles (SOSP)*, 2017, pp. 153–167.

[7] C. Hung, G. Ananthanarayanan, L. Golubchik, M. Yu, and M. Zhang, "Wide-area analytics with multiple resources," in *ACM European Conference on Computer Systems (EuroSys)*, 2018, pp. 1–16.

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

[9] A. Vulimiri, C. Curino, P. B. Godfrey, T. Jungblut, J. Padhye, and G. Varghese, "Global analytics in the face of bandwidth and regulatory

constraints," in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2015, pp. 323–336.

[10] J. Yuan, M. Xu, X. Ma, A. Zhou, X. Liu, and S. Wang, "Hierarchical federated learning through lan-wan orchestration," *arXiv:2010.11612*, pp. 1–11, 2020.

[11] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, B. McMahan *et al.*, "Towards federated learning at scale: System design," in *MLSys*, 2019.

[12] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," in *IEEE Journal of Selected Areas in Communications JSAC*, vol. 37, no. 6, 2019, pp. 1205–1221.

[13] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *arXiv:1610.05492*, 2016, pp. 1–10.

[14] R. Ben Basat, S. Ramanathan, Y. Li, G. Antichi, M. Yu, and M. Mitzenmacher, "Pint: Probabilistic in-band network telemetry," in *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, 2020, pp. 662–680.

[15] Y. Lin, Y. Zhou, Z. Liu, K. Liu, Y. Wang, M. Xu, J. Bi, Y. Liu, and J. Wu, "Netview: Towards on-demand network-wide telemetry in the data center," *Elsevier Computer Networks (CN)*, vol. 180, pp. 1–14, 2020.

[16] Y. Li, H. Lin, Z. Li, Y. Liu, F. Qian, L. Gong, X. Xin, and T. Xu, "A nationwide study on cellular reliability: measurement, analysis, and enhancements," in *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*. ACM, 2021, pp. 597–609.

[17] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *IMLS International Conference on Machine Learning (ICML)*, 2019, pp. 8114–8124.

[18] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," in *International Conference on Learning Representations (ICLR)*, 2020, pp. 1–26.

[19] S. Wang, Y. Ruan, Y. Tu, S. Wagle, C. G. Brinton, and C. Joe-Wong, "Network-aware optimization of distributed learning for fog computing," *IEEE/ACM Transactions on Networking (ToN)*, pp. 1–16, 2021.

[20] Z. Zhou, S. Yang, L. Pu, and S. Yu, "Cefl: online admission control, data scheduling, and accuracy tuning for cost-efficient federated learning across edge nodes," *IEEE Internet of Things Journal (IoTJ)*, vol. 7, no. 10, pp. 9341–9356, 2020.

[21] Y. Guo, Z. Zhao, K. He, S. Lai, J. Xia, and L. Fan, "Efficient and flexible management for industrial internet of things: A federated learning approach," *Elsevier Computer Networks (CN)*, vol. 192, pp. 1–9, 2021.

[22] X. Zhou, X. Dong, Z. Laiping, K. Li, and T. Qiu, "Learning-driven cloud resource provision policy for content providers with competitors," *IEEE Transactions on Cloud Computing (TCC)*, pp. 1–12, 2020.

[23] S. Liu, S. Xue, J. Wu, C. Zhou, J. Yang, Z. Li, and J. Cao, "Online active learning for drifting data streams," *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, pp. 186–200, 2021.

[24] P. Sun, Z. Guo, S. Liu, J. Lan, J. Wang, and Y. Hu, "Smartfct: Improving power-efficiency for data center networks with deep reinforcement learning," *Elsevier Computer Networks (CN)*, vol. 179, pp. 1–9, 2020.

[25] Y. Zhao, C. Tian, J. Fan, T. Guan, X. Zhang, and C. Qiao, "Joint reducer placement and coflow bandwidth scheduling for computing clusters," *IEEE/ACM Transactions on Networking (ToN)*, vol. 29, no. 1, pp. 438–451, 2020.

[26] M. P. Contributors, "An instant virtual network on your laptop (or other pc)." [Online]. Available: http://mininet.org/

[27] D. Pariag and T. Brecht, "Application bandwidth and flow rates from 3 trillion flows across 45 carrier networks," in *Springer PAM*, 2017, pp. 129–141.

[28] K. upon Tensorflow, "Keras recurrent layers." [Online]. Available: https://keras.io/api/layers/recurrent_layers/lstm/

[29] Tensorflow, "Tensorflow federated learning." [Online]. Available: https://www.tensorflow.org/federated/federated_learning

[30] "MNIST," 2021. [Online]. Available: http://yann.lecun.com/

[31] Y. Lin, Y. Zhou, Z. Liu, K. Liu, Y. Wang, M. Xu, J. Bi, Y. Liu, and J. Wu, "Netview: Towards on-demand network-wide telemetry in the data center," *Elsevier Computer Networks (CN)*, vol. 180, p. 107386, 2020.

[32] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "Dïot: A federated self-learning anomaly detection system for iot," in *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 756–767.

[33] M. Calder, X. Fan, Z. Hu, E. Katz-Bassett, J. S. Heidemann, and R. Govindan, "Mapping the expansion of google's serving infrastructure," in *ACM/SIGCOMM Internet Measurement Conference (IMC)*, K. Papagiannaki, P. K. Gummadi, and C. Partridge, Eds., 2013, pp. 313–326.

[34] B. Schlinker, H. Kim, T. Cui, E. Katz-Bassett, H. V. Madhyastha, Í. Cunha, J. Quinn, S. Hasan, P. Lapukhov, and H. Zeng, "Engineering egress with edge fabric: Steering oceans of content to the world," in *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, 2017, pp. 418–431.

[35] M. Calder, X. Fan, Z. Hu, E. Katz-Bassett, J. Heidemann, and R. Govindan, "Mapping the expansion of google's serving infrastructure," in *ACM/SIGCOMM Internet Measurement Conference (IMC)*, 2013, pp. 313–326.

[36] B. Schlinker, H. Kim, T. Cui, E. Katz-Bassett, H. V. Madhyastha, I. Cunha, J. Quinn, S. Hasan, P. Lapukhov, and H. Zeng, "Engineering egress with edge fabric: Steering oceans of content to the world," in *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, 2017, pp. 418–431.

[37] Z. Guo, J. Duan, and Y. Yang, "On-line multicast scheduling with bounded congestion in fat-tree data center networks," *IEEE Journal of Selected Areas in Communications (JSAC)*, vol. 32, no. 1, pp. 102–115, 2013.

[38] C. Kim, A. Sivaraman, N. Katta, A. Bas, A. Dixit, and L. J. Wobker, "In-band network telemetry via programmable dataplanes," in *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, vol. 15, 2015, pp. 1–2.

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

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

[41] T. Chen, Q. Ling, and G. B. Giannakis, "An online convex optimization approach to proactive network resource allocation," *IEEE Transactions on Signal Processing (TSP)*, vol. 65, no. 24, pp. 6350–6364, 2017.

[42] A. Bernstein, E. Dall'Anese, and A. Simonetto, "Online primal-dual methods with measurement feedback for time-varying convex optimization," *IEEE Transactions on Signal Processing (TSP)*, vol. 67, no. 8, pp. 1978–1991, 2019.

[43] R. Gandhi, S. Khuller, S. Parthasarathy, and A. Srinivasan, "Dependent rounding and its applications to approximation algorithms," *Journal of the ACM (JACM)*, vol. 53, no. 3, pp. 324–360, 2006.

[44] A. Mokhtari, A. Ozdaglar, and S. Pattathil, "A unified analysis of extra-gradient and optimistic gradient methods for saddle point problems: Proximal point approach," in *PMLR International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020, pp. 1497–1507.

[45] C. Daskalakis, S. Skoulakis, and M. Zampetakis, "The complexity of constrained min-max optimization," in *ACM SIGACT Symposium on Theory of Computing (STOC)*, 2021, pp. 1466–1478.

[46] B. Kretzu and D. Garber, "Revisiting projection-free online learning: the strongly convex case," in *PMLR International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021, pp. 3592–3600.

[47] A. Koppel, F. Y. Jakubiec, and A. Ribeiro, "A saddle point algorithm for networked online convex optimization," in *IEEE Transactions on Signal Processing (TSP)*, 2015.

[48] Y. Ji, S. Gong, and W. Wang, "Constructing cospectral bipartite graphs," *Elsevier Discrete Mathematics (DM)*, vol. 343, no. 10, pp. 1–7, 2020.

[49] J. Cao, X. Lin, S. Guo, L. Liu, T. Liu, and B. Wang, "Bipartite graph embedding via mutual information maximization," in *ACM International Conference on Web Search and Data Mining (WSDM)*, 2021, pp. 635–643.

[50] "AWS Cloud Instance Comparision," 2017. [Online]. Available: https://www.cloud omatic.com/aws-gcp-comparison/

[51] "AMPL," 2021. [Online]. Available: https://ampl.com/

[52] "IPOPT," 2021. [Online]. Available: https://github.com/coin-or/Ipopt/

[53] L. Zhang, C. Wu, Z. Li, C. Guo, M. Chen, and F. C. Lau, "Moving big data to the cloud: An online cost-minimizing approach," *IEEE Journal of Selected Areas in Communications (JSAC)*, vol. 31, no. 12, pp. 2710–2721, 2013.

[54] X. Yang, H. Lin, Z. Li, F. Qian, X. Li, Z. He, X. Wu, X. Wang, Y. Liu, Z. Liao, D. Hu, and T. Xu, "Mobile access bandwidth in practice: measurement, analysis, and implications," in *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*. ACM, 2022, pp. 114–128.

[55] V. Smith, S. Forte, C. Ma, M. Takáč, M. I. Jordan, and M. Jaggi, "CoCoA: A general framework for communication-efficient distributed

optimization," in *Journal of Machine Learning Research (JMLR)*, vol. 18, no. 1, 2017, pp. 8590–8638.

[56] R. Viswanathan, G. Ananthanarayanan, and A. Akella, "Clarinet: Wan-aware optimization for analytics queries," in *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2016, pp. 435–450.

[57] L. Chen, J. Cao, H. Tao, and J. Wu, "Trip reinforcement recommendation with graph-based representation learning," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, pp. 1–20, 2022.

[58] J. Meng, H. Tan, C. Xu, W. Cao, L. Liu, and B. Li, "Dedas: Online task dispatching and scheduling with bandwidth constraint in edge computing," in *IEEE Conference on Computer Communications (INFOCOM)*, 2019, pp. 2287–2295.

Yibo Jin received the BS degree, the Elite Program, from the Department of Computer Science and Technology, Nanjing University in 2017. He is pursuing the PhD degree under the supervision of Prof. Sanglu Lu. He was a visiting student with the Hong Kong Polytechnic University, Hong Kong in 2017, under the supervision of Prof. Song Guo. To date, he has published more than 15 papers, in journals such as JSAC, ToN and TPDS, and in conferences such as INFOCOM, ICDCS, ICPP, SECON and IWQoS. He received the Best Paper Candidates of IEEE WoWMoM 2021. His research interests include big data analytics and edge computing. He is the student members of the IEEE, the ACM and the CCF.

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 Science, University of Oregon, USA. Previously he worked as a member of technical staff at 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 is a recipient of the NSF CAREER award. 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 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 has been on the program committees of INFOCOM, MOBIHOC, ICDCS, TheWebConf, and IWQoS, and served as the program chair of multiple workshops with INFOCOM and ICDCS.

Mingtao Ji received the BS degree from the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics of in 2018. He is currently pursuing the PhD degree under the supervision of Professor Zhuzhong Qian in Nanjing University. To date, he has already published over 5 papers, including in journals such as Electric Power ICT, and in conferences such as ISPA. His research interests include big data analytics and distributed machine learning.

Zhuzhong Qian is a professor at the Department of Computer Science and Technology, Nanjing University, P. R. China. He received his PhD. Degree in computer science in 2007. Currently, his research interests include cloud computing, distributed systems, and pervasive computing. He is the chief member of several national research projects on cloud computing and pervasive computing. He has published more than 30 research papers.

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.

Ning Chen is currently working toward the PhD degree in the Department of Computer Science and Technology, Nanjing University, under the supervision of Prof. Sheng Zhang. His research interests include edge computing, deep reinforcement learning, and video streaming. His publications include those appeared in IEEE TPDS, SECON, ICPADS and Computer Network.

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.