Placement of Parameter Server in Wide Area Network Topology for Geo-Distributed Machine Learning

Yongyao Li, Chenyu Fan, Xiaoning Zhang, and Yufeng Chen

***Abstract*—Machine learning (ML) is extensively used in a wide range of real-world applications that require data all around world to pursue high accuracy of a global model. Unfortunately, it is impossible to transmit all gathered raw data to a central data center for training due to data privacy, data sovereignty and high communication cost. This brings the idea of geo-distributed machine learning (Geo-DML), which completes the training of the global ML model across multiple data centers with the bottleneck of high communication cost over the limited wide area networks (WAN) bandwidth. In this paper, we study on the problem of parameter server (PS) placement in PS architecture for communication efficiency of Geo-DML. Our optimization aims to select an appropriate data center as the PS for global training algorithm based on the communication cost. We prove the PS placement problem is NP-hard. Further, we develop an approximation algorithm to solve the problem using the randomized rounding method. In order to validate the performance of our proposed algorithm, we conduct large-scale simulations, and the simulation results on two typical carrier network topologies show that our proposed algorithm can reduce the communication cost up to 61.78% over B4 topology and 21.78% over Internet2 network topology.**

***Index Terms*—Geo-distributed machine learning, routing, wide area networks.**

I. Introduction

MACHINE learning (ML) is a method of data analysis, which extracts helpful information from large-scale data to make predictions or decisions without human intervention [1]. Nowadays, ML algorithms have been adopted in many classes of applications such as computer vision [2], speech recognition [3], and natural language processing [4], etc., where it is difficult to use conventional methods to complete complicated tasks. These applications need to collect and analyze a variety of data including user behaviors, videos, pictures and voices, etc., which are generated at a very fast speed from users of all over the world. Many Internet service providers (ISPs), such as Google [5], Microsoft [6], and Amazon [7], have tens of data centers around the world to reduce their services' access delay for users, and to collect large-scale data.

A frequently-used approach to train a ML model over vast data is to transmit all data into one data center by wide area networks (WAN). However, this approach faces the following three challenges. The first challenge is high communication cost. The link bandwidth of WAN is seriously limited, and transferring such an enormous amount of data to a data center takes a long time, which undoubtedly results in high communication cost for ML training. The second challenge is data privacy. The ML training data possibly comes from different ISPs. It is difficult to completely share the raw data in consideration of data security. The third one is data sovereignty. Nowadays, an increasing number of countries enact laws to prohibit the data transmission across national or continental borders [8], [9].

To address these above challenges, the concept of geo-distributed machine leaning (Geo-DML) is proposed in industry and academia [10], [11]. In the paradigm of Geo-DML, a large amount of raw data is locally stored in different data centers, and the ML model training procedure is performed over the geographically dispersed data sets among data centers via the communication of WAN links. However, the existing DML algorithms are suitable for the server cluster with high-speed local area networks (LAN) (e.g., 10 gigabit ethernet) within a single data center. In the context of Geo-DML, the bandwidth of WAN is much smaller than that of high-speed LAN, which causes a slowdown for DML parameter synchronization. In addition, the link bandwidth of WAN connecting data centers are mainly determined by the geographic distance. For example, the link bandwidth between geographically-close data centers could be up to 12 times higher than that between distant ones [12]. Therefore, how to accelerate the training procedure by improving communication efficiency in Geo-DML is an important issue to be studied.

In the study we mainly consider the parameter server (PS) architecture for Geo-DML. There are two kinds of computing nodes in traditional PS architecture, including several workers which train ML model replicas on data shards in parallel, and a PS which is responsible for synchronizing these model

Manuscript received May 12, 2022 revised September 26, 2022; approved for publication by Hongjian Sun, Division 3 Editor, May 9, 2023.

This work was supported in part by the Natural Science Foundation of Sichuan (No. 2022NSFSC0543), in part by the National Natural Science Foundation of China (NSFC) (No. 62001087, 62171085, 61871097, 62272428, and U20A20156), in part by Hefei Municipal Natural Science Foundation (No. 2022004), in part by Anhui Provincial Natural Science Foundation (No. 2208085MF167), in part by the Open Research Projects of Zhejiang Lab (No. 2021LC0AB04), and in part by Natural Science Foundation of Jiangsu Province (No. BK20221261).

Y. Li and Y. Chen are with Macau University of Science and Technology Ringgold standard institution, Macau, China, email: {yongyaoli1980, chyf01}@163.com.

C. Fan and X. Zhang are with University of Electronic Science and Technology of China Ringgold standard institution - School of Information and Communication Engineering, Chengdu, China, email: fancy4528@gmail.com, xnzhang@uestc.edu.cn.

Y. Li is the corresponding author.

Digital Object Identifier: 10.23919/JCN.2023.000021

1229-2370/23/$10.00 © 2023 KICS

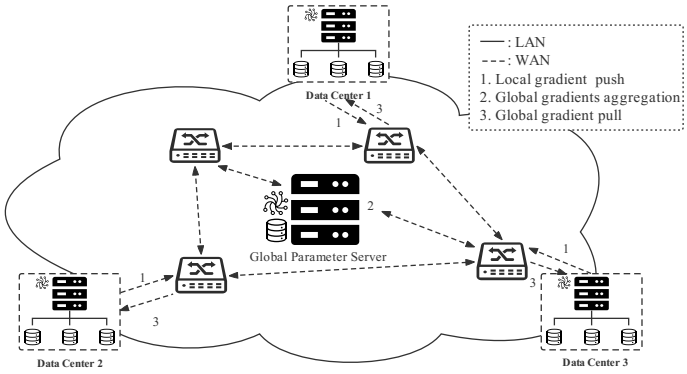


Fig. 1. Centralized hierarchical geo-distributed machine learning structure.

replicas. In the procedure of training, each worker in PS architecture calculates parameter updates (i.e., gradients) and then pushes them to the PS. The global model will not be updated until the PS have received gradients from all workers in each epoch. Then each worker starts the next-epoch training after pulling the latest model parameters.

In the context of Geo-DML, traditional PS architecture is usually hierarchical constructed to avoid severe network congestion over WAN. A typical centralized hierarchical Geo-DML structure is shown in Fig. 1. There are some worker data centers and a central data center as the global PS in the system. Within each worker data center, there are several worker nodes and a local PS. Each worker node contains the same model replica and a shard of input data samples to achieve data parallelism. The basic training procedure during one epoch is described as following: 1. After worker nodes completing the gradient descent calculation, they push the gradients to the local PS over LAN. The local PS aggregates the received gradients and push the locally aggregated gradient to the global PS over WAN. 2. The global PS aggregates the received gradients. 3. The global PS broadcast the globally aggregated gradient back to each local PS. Local PSs update their models by the globally aggregated gradient and broadcast the updated model parameters to the worker nodes in the corresponding data center for the next round of gradient descent calculation.

In this paper, we study on the problem of PS placement in PS architecture for communication efficiency of Geo-DML, which aims to select an appropriate data center as the PS for the global ML training based on the communication time. Since the PS placement problem is proved NP-hard, we propose an approximation algorithm using the randomized rounding method [13] named approximate parameter server placement (APSP). In the APSP algorithm, we select the appropriate one from all the data centers as the global PS based on the communication time in each epoch. In order to validate the performance of our proposed algorithm, we conduct large-scale simulations, and the simulation results on two typical carrier network topologies show that our proposed algorithm can reduce the communication cost up to 61.78% over B4 topology [30] and 21.78% over Internet2 network topology. As far as we know, our paper is the first study to investigate the PS placement problem in Geo-DML.

The main contributions of our work are summarized as follows.

- We investigate the problem of PS placement in PS architecture for Geo-DML, and formulate the studied problem as an optimization model.
- Since the formulated optimization model is NP-hard, we propose an approximation algorithm using the randomized rounding method with a bounded approximation factor named APSP to solve this problem. Further, we analyze the approximation performance of APSP algorithm and give the approximation factor for link capacity constraint.
- Extensive simulation results on two typical carrier network topologies show that our proposed algorithm can reduce the communication cost up to 18% in average.

The rest of this paper is organized as follows. In Section II, we briefly review the literatures related to our work. In Section III, we give a motivation example of our work and formulate the studied problem as an optimization model. In Section IV, we describe our solution and present our proposed algorithm. We show the simulation results in Section V and give our conclusion in Section VI.

II. Related Work

To analyze the large amount of data with a fast growing rate all around world, distributed training is the effective way to maintain capability and scalability while keeping cost controllable. Recently, some Geo-DML architecture have been proposed for global data analysis across multiple data centers.

Some prior works are aware of the challenge of globalization of model training and establish the arising problem of analyzing the data generated globally in the context of data analytics systems (e.g., [16]–[19], [21], [27]). These works show the very optimistic WAN bandwidth reduction and improvement of system performance using a WAN-aware data analytics framework. For instance, Pu *et al*. [16] presented a system for low latency Geo-Distributed analytics named Iridium, which achieved low query response times by optimizing placement of both data and tasks of the queries. The experiment showed that the Iridium sped up queries by 3×–19× and lowered WAN usage by 15%–64% compared to existing baselines. Vulimiri *et al*. [17] proposed WANalytics, a system that pushed computation to edge data centers, automatically optimizing workflow execution plans and replicating data when needed. Their Hadoop-based prototype delivered 250× reduction in WAN bandwidth on a production workload from Microsoft. Nevertheless, their goal was not to run an ML procedure efficiently on WAN, which has highly dissimilar challenges from data analytics systems.

There are also some works focused on Geo-DML from different perspectives. Gaia [10] first discussed the problem of running an ML system on geo-distributed data and formalized challenges in Geo-DML. They proposed a Geo-DML structure decoupling the communication within a data center from that between data centers, which allowed different communication and consistency models for each data center. They also proposed a synchronization ML model named approximation synchronous parallel (ASP), whose main idea was

dynamically eliminating unimportant communication between data centers with correctness guarantee of ML algorithms. Their experiment showed that Gaia provides 1.8×–53.5× speedup over two state-of-the-art DML systems over WAN. Cano *et al*. [22] proposed a Geo-DML system that offset the generally communication-intensive nature of ML algorithms by employing and extending communication-sparse ones. The experimental evaluation indicated that their approach could outperform other state-of-the-art systems by several orders of magnitude when measuring X-DC transfers, as well as respect stricter sovereignty constraints. WeightGrad [11] was a two level structure (TLS) Geo-DML system built on the frame of PS architecture [14], [15] that was used for communication over both LAN and WAN. For local convergence, WeightGrad provided networks with loss-aware weight quantization and gradient quantization. For global convergence, it utilized the idea of ASP to abandon insignificant communication between data centers while guaranteeing the correctness of deep neural network (DNN) models. Experiments of WeightGrad showed 5.36× speedup over baseline and 1.4×–2.26× speedup compared with the four state-of-the-art DML systems.

Some works studied the routing and scheduling issues in geo-distributed networks. Zhao *et al*. [23] proposed RAPIER, a framework of coflow-aware network optimization that integrated routing and scheduling for better application performance. In their experiments, RAPIER reduced the average coflow completion time by up to 79.30% compared to the scheduling-only solution. [24]–[26] also started to consider coflow level optimization in Geo-Distributed networks. Hung *et al*. [27] presented job scheduling algorithms coordinating job scheduling across data centers with low overhead. It gained 50% improvement in average job completion time over the shortest remaining processing time based approaches. In [28], the heaviest-load-first (HLF) algorithm was proposed for intra-job scheduling and shortest weighted remaining time first (SWRTF) algorithm for inter-job scheduling. In the simulations, HLF reduced the iteration communication time by 64.97% compared to the circuit scheduler Sunflow, and SWRTF saved 42.9%, 54.2%, and 27.2% of weighted job completion time compared to shortest-job-first, Baraat and weighted-first algorithms, respectively.

Though these works focused on certain aspects of Geo-DML, there is still a lot of room for improvement in reducing the communication cost. In the general architecture of Geo-DML, there will be a global PS to execute the global aggregation. Since the data centers participating in training are geo-distributed over WAN, a proper selection of the global PS location can benefit the reduction of communication time during the training procedure. To our knowledge, none of the prior works considered about the placement of PS. Our work, APSP, enhances communication efficiency of Geo-DML by optimizing PS placement in PS architecture.

III. Motivation and Problem Formulation

In this section, we first show a motivation example of our work. Then we give a substrate geo-distributed network, followed by the formulation of the optimization model describing the PS placement problem. For clear presentation, all the notations used in this paper are summarized in TABLE I.

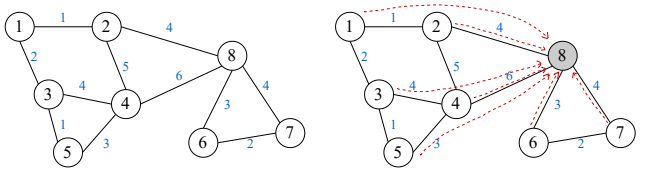
TABLE I
NOTATIONS USED IN THE PAPER.

Notation	Description
$G(V, E)$	The network topology, where V denotes the set of nodes and E denotes the set of links
$e(u, v)$	The physical link between node u and node v, where $u, v \in V$, $e(u, v) \in E$
B_e	The total available bandwidth of link e
F	The set of data flows, with $f \in F$
b_f	The available bandwidth allocated to data flow f, where $f \in F$
P_f	The path set of data flow f, with $p \in P_f$, where $f \in F$
δ_{efp}	A constant. It takes 1 if link e belongs to path p of data flow f, 0 otherwise, where $f \in F$, $p \in P_f$
y_v	A binary variable. It takes 1 if node v is selected as the PS, 0 otherwise, where $v \in V$
x_{fp}	A binary variable. It takes 1 if path p is chosen for data flow f to transmit parameters, 0 otherwise, where $f \in F$, $p \in P_f$
$C_v(u)$	The communication time between node u and PS v
m	The variable denoting $1/T$
z_{fp}	The variable denoting $m \cdot x_{fp}$
N	The number of worker DCs
Ep	The threshold of epoch number
D^i	Dataset of the ith worker data center
D_t^i	Mini-batch of D_i used in the tth epoch
$\boldsymbol{s}$	Sample drawn from the dataset
$\boldsymbol{w}$	d dimensional parameter vector of ML model, $\boldsymbol{w} \in \mathbb{R}^d$
$\boldsymbol{w}_t$	The parameter in the tth epoch
$\boldsymbol{w}_t^i$	The parameter of worker DC i in the tth epoch
$l(\boldsymbol{w}, \boldsymbol{z})$	The loss function of the model with parameter $\boldsymbol{w}$ in data point $\boldsymbol{s}$
η_t	The learning rate in the tth epoch
$\boldsymbol{g}$	The general gradient calculated as $\boldsymbol{g} = \nabla_{\boldsymbol{w}} l(\boldsymbol{w}, \boldsymbol{s})$
$\boldsymbol{g}_t^i$	The accumulated gradient of worker data center i in the tth epoch
$\bar{\boldsymbol{g}}_t$	The aggregated gradient in the tth global epoch

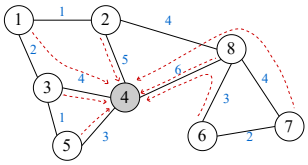
A. Motivation

Compared to the communication cost of global gradient exchanging, the communication time of local gradient exchanging in worker data centers via high-speed LAN environment is not our optimization target. Hence, our work mainly focuses on the reduction of communication cost while transmitting the gradients globally by determining which data center to be the global PS. For simplicity, each data center is abstracted as a node in the WAN topology.

Fig. 2 shows a motivation example of our approach. Consider an eight-node WAN in Fig. 2(a), where the numbers in blue beside each link denote the available bandwidth of this link. Assume that all the worker nodes start gradient transmission at the same time and the data size remains the same in each epoch. As shown in Fig. 2(b), node 8 is randomly selected as the PS and the paths are prearranged for every node pair in plan A. The straggler node is node 1, whose available bandwidth to the PS (node 8) is 1. Fig. 2(c) illustrates that in plan B, node 4 is selected as PS by our proposed algorithm, which also determines the paths for each data flow at the same time. The straggler node is still node 1, while its available bandwidth to the PS (node 4) is 2 in this case. Obviously, the



(a) Original topology (b) Plan A: Node 8 as the PS



(c) Plan B: Node 4 as the PS

Fig. 2. The motivation example of geo-DML in WAN topology.

communication time of the gradient transmission in plan B will be shorter than that in plan A during one epoch.

B. Network Model

The substrate network information is already known in our study. We use $G(V, E)$ to model the substrate network, where V denotes the set of nodes in the substrate network, and E denotes the set of links. $e(u, v)$ is the link between node u and node v, where $u, v \in V$. For each link $e \in E$, we use B_e to denote the available bandwidth of link e.

In our study, we use F to denote the set of the worker-to-server data flows and b_f to denote the available bandwidth allocated to data flow f that belongs to F. Because the data size that each worker node transmits to the PS is fixed and same, we assume that the transmitting data size of each data flow $f \in F$ is D. We use P_f to denote the set of feasible paths from the source node to the destination node of the flow f, with $p \in P_f$. δ_{efp} is a constant to denote whether link e belongs to path p of data flow f. $\delta_{efp} = 1$ if link e on the path p, 0 otherwise.

C. Problem Formulation

We formulate the PS placement problem as an optimization model. Our objective is to minimize the overall communication time per epoch during training by appropriate PS selection. It is apparently that when the topology keeps the same, if the communication time of worker-to-server data flows has been minimized, the communication time of server-to-worker data flows would also be minimized. Thus, we can simplify the minimization objective as twice the worker-to-server communication time per epoch.

Firstly, we use a binary variable y_v to denote the location of PS. If node v is the PS, $y_v = 1$; otherwise, $y_v = 0$. Since only one node can be chosen as a PS at one time, we derive the following constraint:

$$\sum_{v \in V} y_v = 1. \tag{1}$$

Another binary variable x_{fp} is used to denote whether the path p is chosen for the data flow f to transmit parameters. $x_{fp} = 1$, if path p is chosen for data flow f; $x_{fp} = 0$, otherwise. In this scenario, each worker node can only choose one path for parameter transmission. Hence, we give the following constraint:

$$\sum_{p \in P_f} x_{fp} = 1, \forall f \in F. \tag{2}$$

Different data flows may pass through the same link e. Due to the limitation of bandwidth of link e, we can derive the following link bandwidth constraint:

$$\sum_{f \in F} \sum_{p \in P_f} \delta_{efp} x_{fp} b_f \leq B_e, \forall e \in E. \tag{3}$$

When node v is selected as the PS, let the data flow between worker node u and PS be f_u. In fact, the upload bandwidth and download bandwidth of a path are not exactly the same with small difference. However, the relationship of size of bandwidth remains the same over different paths, whether upload or download. Therefore, we assume that a path has the same upload bandwidth and download bandwidth for simplicity, which will not affect the final result. Based on this assumption, we can compute the communication time of the transmission as:

$$C_v(u) = 2 \sum_{p \in P_{f_u}} x_{f_u p} \cdot |p| \frac{D}{b_{f_u}}, \tag{4}$$

where $|p|$ is the number of hops in path p. Note that we omit the influence of propagation delay in this work, since the transmission delay is two orders of magnitude greater than the propagation delay.

With the assumption that the transmissions from worker nodes occur simultaneously, we can represent the communication time in an epoch as:

$$\max_{u \neq v}(C_v(u)) \tag{5}$$

Synthesize the above constraints, we formulate the PS placement problem as an optimization model:

$$\text{minimize} \sum_{v \in V} y_v \cdot \max_{u \neq v}(C_v(u)) \tag{6}$$

subject to: (1), (2), (3),

$$x_{fp} \in \{0, 1\}, \forall f \in F, \forall p \in P_f, \tag{6a}$$

$$y_v \in \{0, 1\}, \forall v \in V, \tag{6b}$$

$$b_f \geq 0, \forall f \in F. \tag{6c}$$

Equation (6) is used to denote the optimization model of the PS placement problem.

The key to solve the above problem is to figure out the minimum value of $\max_{u \neq v}(C_v(u))$ with a fixed v, which is the global communication time of one epoch with v being the PS. Hence, we can easily determine the PS node by comparing the corresponding communication time. Consequently, we can focus on the following subproblem to work out the minimum communication time with a fixed v:

Problem 1:

$$\text{minimize } T \tag{7}$$

subject to: (2), (3), (6a), (6c),

$$T = \max_{u \neq v} C_v(u). \tag{7a}$$

We use *Problem 1* to denote the optimization model of the above subproblem in (7). Solving *Problem 1* helps us to determine which path that each flow will choose by x_{fp} and determine the bandwidth allocated to each flow by b_f, as well as figure out the communication time.

Theorem 1. *Problem 1 is NP-hard.*

Proof. Since the aggregating flows can be viewed as a coflow, we can consider *Problem 1* as a special case of a single coflow routing and scheduling problem whose flows have the same destination. It is impractical to find the optimal solution of single coflow problem due to the nonlinear form and the existence of binary variables. The single coflow problem is an integer multi-commodity flow problem that is proven to be NP-hard [29]. Therefore, the original *Problem 1* is also NP-hard. □

D. Relaxed Formulation

A NP-hard problem is intractable to be solved theoretically, hence we consider to design an efficient approximation algorithm based on randomized rounding to solve *Problem 1*. Notice that some constraints of *Problem 1* is not a standard linear form. Thus, we first introduce a new variable m to denote $1/T$. Then we can rewrite *Problem 1* as follows:

Problem 2:

$$\text{maximize } m \tag{8}$$

subject to: (2), (3), (6a), (6c),

$$m \leq \frac{1}{C_v(u)}, \forall u \in V, u \neq v. \tag{8a}$$

Problem 2 is used to denote (8).

To transform *Problem 2* into a standard linear programming (LP) problem, the binary variable x_{fp} needs to be relaxed to real numbers in the range 0 to 1. Here we have the *Problem 3*:

Problem 3:

$$\text{maximize } m \tag{9}$$

subject to: (2), (3), (6c), (8a),

$$x_{fp} \in [0,1], \forall f \in F, \forall p \in P_f. \tag{9a}$$

Now there is still one last nonlinear constraint. In order to merge two variables (i.e., b_f and x_{fp}) into one, we introduce a new variable $z_{fp} = m \cdot x_{fp}$. From existing conditions, it is easy to draw the conclusion that $m \cdot x_{fp} \leq b_f \cdot x_{fp}/|p| \cdot D$, thus $|p| \cdot D \cdot z_{fp} \leq b_f \cdot x_{fp}$. We substitute z_{fp} into *Problem 3* and obtain the standard LP model (*Problem 4*):

Problem 4:

$$\text{maximize } m \tag{10}$$

subject to:

$$\sum_{f \in F} \sum_{p \in P_f} \delta_{efp} z_{fp} \cdot |p| \cdot D \leq B_e, \forall e \in E, \tag{10a}$$

$$z_{fp} \geq 0, \forall f \in F, \forall p \in P_f, \tag{10b}$$

$$\sum_{p \in P_f} z_{fp} = m, \forall f \in F. \tag{10c}$$

Now z_{fp} and m are variables of *Problem 4* which can be efficiently solved by standard LP solvers, such as GUROBI and CPLEX. However, the binary variable x_{fp} obtained according to z_{fp} and m is relaxed and may be fractional, which is not a feasible solution of *Problem 2*. Therefore, we propose a randomized rounding based algorithm to find a feasible solution, shown in next section.

IV. Algorithm Design

In this section, we propose a randomized rounding based approximation algorithm named APSP to solve the complexity of the key subproblem of the PS placement problem. In our design, we first formulate a model (6) based on the Geo-DML job flows and the network topology. Next, we extract the key subproblem as an optimization model *Problem 1* from (6) and relax it into a LP model *Problem 4*. After solving *Problem 4* by an LP solver (e.g., GUROBI and CPLEX), we utilize randomized rounding technique to enforce the solution of *Problem 4* to be a feasible one of *Problem 1*. The basic idea of our algorithm is to select the path of each flow according to the rounding solution of *Problem 4* firstly, and then to guide available bandwidth assigned to flows based on the selected paths.

Firstly, we describe the algorithm of the total training procedure as Algorithm 1 omitting the calculation and aggregation within the worker data center. Before the training, we choose an arbitrary worker node being aware of the global topology and utilize APSP on it to determine the global PS v and path set P of the training, which are broadcast to all data centers. For each worker data center i, the general training program is executed on it (line 1 to line 7). Worker data center initialize the model by the weight $\boldsymbol{w_i}$ pulled from PS (line 1). Worker data center i calculates the local gradient with the mini-batch D_t^i of the data shard D^i and push the local gradient to the PS (line 3 and line 4). Then parameters of the local model in worker data center i are updated by the aggregated gradient pulled from PS for the next epoch (line 5 and line 6). As for the PS, it initializes the model parameters $\boldsymbol{w}$ randomly and broadcasts the initial parameters to all the worker data centers (line 1 and line 2). The PS gathers the gradients from worker data centers (line 4 to line 6). The PS calculates the aggregated gradient $\bar{\boldsymbol{g}}_t$ and push it to worker data centers (line 7 and line 8).

The APSP is summarized as Algorithm 2, whose time complexity is $O(NM+N^2)$ and space complexity is $O(NE)$, where M denotes the time complexity of the method for solving LP problem and E denotes the number of links in the network. We go through every node v in set V and solve the subproblem to get the communication time of the case that

Algorithm 1 Geo-DML PS system with APSP

Preparatory work on a arbitrary worker node

1: Find PS v and path set P by APSP;
2: Broadcast v, P to All data centers;

Worker Data Center i **:** $i \neq v, i = 1, \cdots, N$

Input: Ep, D^i, p.
1: Pull $\boldsymbol{w}_i$ from PS through path p;
2: **for** $t \in Ep$ **do**
3: Calculate gradient $\boldsymbol{g}_t^i = \frac{1}{|D_t^i|} \sum_{\boldsymbol{s} \in D_t^i} \nabla_{\boldsymbol{w}_t^i} l(\boldsymbol{w}_t^i, \boldsymbol{s})$;
4: Push $\boldsymbol{g}_t^i$ to PS through path p;
5: Pull $\bar{\boldsymbol{g}}_t$ from PS through path p;
6: Update Parameters $\boldsymbol{w}_{t+1}^i \leftarrow \boldsymbol{w}_t^i - \eta_t \cdot \bar{\boldsymbol{g}}_t^i$;
7: **end for**

Parameter server v

Input: Ep, N.
1: $\boldsymbol{w} \leftarrow$ initial parameter values (weights);
2: Broadcast $\boldsymbol{w}$ to worker data centers;
3: **for** $t \in Ep$ **do**
4: **for** $i \in N$ **do**
5: Pull $\boldsymbol{g}_t^i$ from worker data center i;
6: **end for**
7: Aggregate gradients $\bar{\boldsymbol{g}}_t = \sum_i \boldsymbol{g}_t^i / N$;
8: Push $\bar{\boldsymbol{g}}_t$ to worker data center i;
9: **end for**

Algorithm 2 Approximate parameter server placement (APSP)

Input: $G, E, V, B_e, F, P_f, \delta_{efp}$.
Output: $v^*, x_{fp}^*, b_f^*, T^*$
1: Initialize $T^* \leftarrow \inf$;
2: **for** $v \in V$ **do**
3: Initialize $x_{fp} = 0, \forall f \in F, \forall p \in P_f$;
4: $z_{fp}, m =$solve Problem(4);
5: $\hat{x}_{fp} = \frac{z_{fp}}{m}$;
6: **for** $f \in F$ **do**
7: **for** $p \in P_f$ **do**
8: $x_{fp} = Bernoulli(possibility = \hat{x}_{fp})$;
9: **if** $x_{fp} = 1$ **then**
10: $x_{fp'} = 0, p' \in P_f, p' \neq p$;
11: break;
12: **end if**
13: **end for**
14: **end for**
15: $b_f, m' =$solve Problem(2) according to x_{fp};
16: $T \leftarrow \frac{1}{m'}$;
17: **if** $T < T^*$ **then**
18: $T^* \leftarrow T$;
19: $v^* \leftarrow v$;
20: $x_{fp}^* \leftarrow x_{fp}, \forall f \in F, \forall p \in P_f$;
21: $b_f^* \leftarrow b_f, \forall f \in F$;
22: **end if**
23: **end for**
24: return $v^*, x_{fp}^*, b_f^*, T^*$;

select v as the PS. Node v with the shortest communication time is determined as the PS v^* of next epoch. We will describe the subproblem in detail (line 2 to line 23). First, we initialize all x_{fp} to 0 (line 3). Next, we get all z_{fp} and m by solving *Problem 4* and utilize them to calculate the fractional $\widehat{x}_{fp}$ (line 4 and line 5). Then we select the only path p for the flow f by set the corresponding x_{fp} to 1 according to randomized rounding (line 6 to line 14). Afterwards we can solve *Problem 2* with fixed binary x_{fp} and obtain available bandwidth allocated to each flow b_f and the new m' (line 15). Then, we calculate the communication time of a single epoch T for PS communication structure with fixed PS v (line 16). Finally, we select the PS node based on the communication time and output the chosen node, along with the path set and bandwidth allocation (line 17 to line 24).

It is necessary to introduce the randomized rounding technique first, which is the key technique of path selection process. Randomized rounding is a technique designed to solve 0-1 integer linear programming (ILP) problems. It can transform an optimal solution of the corresponding relaxed problem into a probably good solution of the original 0-1 ILP problem and provide bounds on the disparity between the rational and 0-1 optima for a given problem instance.

The general outline of randomized rounding can be described as follows. Assume Γ_I is a 0-1 ILP, with binary variables $x_i \in \{0, 1\}$ and Γ_R is the rational relaxation of Γ_I, with $x_i \in [0, 1]$. The basic algorithm is composed of the following two phases: (1) Solve Γ_R and let the variables be assigned values $\hat{x}_i \in [0, 1]$; (2) set the variables x_i randomly to 0 or 1 according to the rule $\Pr[x_i = 1] = \hat{x}_i$.

We analyze the probability bound of violating link bandwidth constraints after rounding as follows:

Theorem 2. *Our algorithm achieves the approximation factor of* $(2 \ln n + 2\sqrt{\ln^2 n + 2\alpha \ln n})/\alpha + 1$ *after randomized rounding for link bandwidth constraints.*

The proof of Theorem 2 is shown in Appendix A.

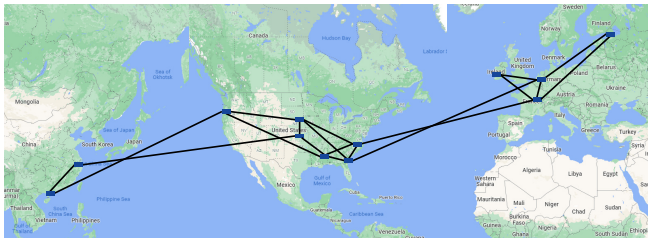
The above discussed APSP in the theoretical case, while APSP can be extended for more general scenarios. For example, consider that the available bandwidth can change from time to time in practice, APSP can be used multiple times to change the PS node dynamically. Since APSP is an algorithm to solve LP problem, the solving time will not be longer than the time saved by proper PS selection.

V. Performance Evaluation

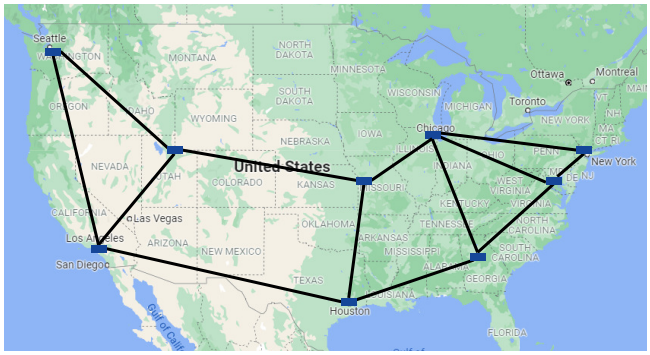
In this section, we evaluate our proposed algorithm through theoretical analysis. We present our simulation settings first, then we show the simulation results of the proposed algorithm.

A. Simulation Setting

Network topology: We use two real-world inter data center networks to evaluate the performance of our proposed algorithm, including B4 [30] topology and Internet2 network topology. B4 is the Googles private WAN, which contains 12 data centers all around the world connected by 19 inter



(a) B4 Topology



(b) Internet2 Topology

Fig. 3. Real world topologies used in our work.

TABLE II
PARAMETERS OF DNN MODELS.

Precision	Model	Parameters	Total size
32 bits/4bytes	ResNet50	23 M	92 MB
	ResNet152	60 M	240 MB
	AlexNet	64 M	256 MB
	VGG-Net	138 M	552 MB

data center links as Fig. 3(a) shows. The Internet2 network topology is illustrated in Fig. 3(b), which has 9 data centers and 13 inter data center links across the United States. For simplicity, we set the available bandwidth of each undirected link in the range of 500 Mbps to 5 Gbps according to the physical distance of the link.

Job workloads: To simulate the Geo-DML job workloads, we assume the data size that the worker data center transmits to the PS in one epoch is in range of 92 MB to 256 MB. The data size is the number of parameters of some state-of-the-art convolutional neural networks (CNN) training models actually in use, including ResNet50, ResNet152 [31], AlexNet [32] and VGG-Net [33]. Table II shows the size and number of parameters in full precision (i.e., 32 bits/4bytes) of the above CNN models.

Solutions to compare: Our proposed algorithm solves the PS selecting and routing problem at once. Consider that as far as we know, there is no existing method covering two aspects. Therefore, we compare our algorithm in different aspects with the following solutions (compare with the former two to demonstrate the performance of PS selecting, latter two to show the performance of routing):

- **Randomly selected:** Select the PS node randomly and will not change throughout the training.
- **Most connected:** Select the PS node that has the most links connected to it and will not change throughout the training.
- **Shortest path routing:** Choose the shortest path for each flow with the fixed PS node.
- **Load balancing routing:** Choose paths for flows that can balance the load with fixed PS node.

Performance metrics: In the context of Geo-DML, we only focus on the communication time in one epoch, which denotes the global model synchronization communication time of a jobs single epoch.

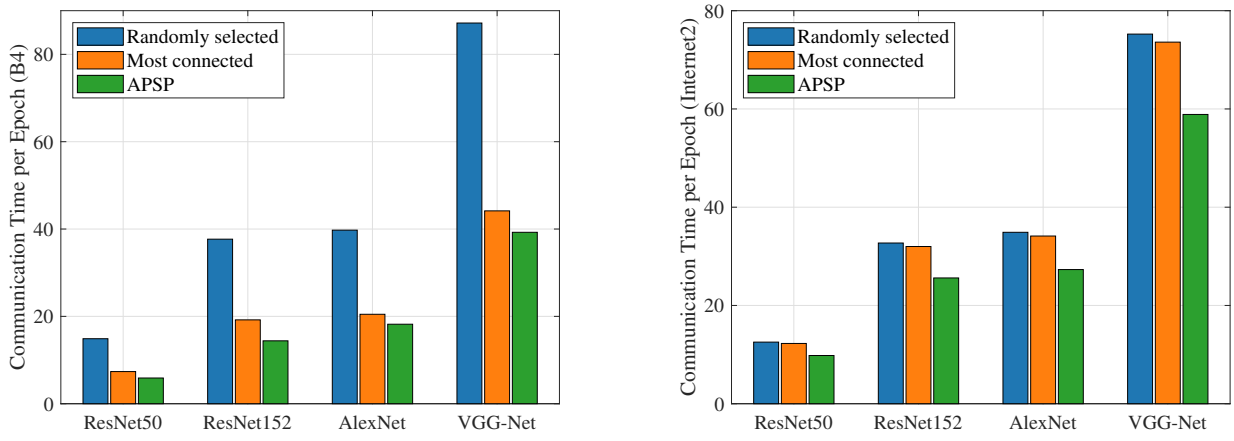
B. Simulation Results

We investigate the performance of APSP by comparing it to cases that randomly select the PS and select the most connected node as the PS over B4 and Internet2 network topologies. The experimental results and analysis are presented below.

Fig. 4 shows the one-epoch communication time of three cases (i.e., randomly selecting the PS, selecting the most connected node as the PS and selecting the PS by APSP) using four state-of-art CNN including ResNet50, ResNet152, AlexNet and VGG-Net. Fig. 4(a) is the results on B4 topology while Fig. 4(b) is the results on Internet2 network topology. It is evident that the performance of ASPS is always better than the other two solutions because of the proper location of the PS. In addition, the communication time shows an upward trend with the CNN model expansion, since the size of transmitted data grows as the model scale grows.

Table III shows the numerical results of our experiments. It can be calculated from it that APSP reduces the communication time by 52.40%–61.78% over the solution that randomly selects the PS and by 11.10%–25.00% over the solution that selects the most connected PS on B4 topology. As for Internet2 network topology, ASPS reduces the communication time by 21.69%–21.78% and by 19.90%–21.04% over the solution that randomly selects the PS and selects the most connected PS respectively. Though the most connected node may be the proper PS location with higher probability than a random node, the node selected by APSP is determined to be the optimal node in the current topology. Thus, APSP obtains less reduction of communication time over the solution that selects the most connected PS, compared with random selection. Although the speedup varies from topologies, APSP accelerate the communication during Geo-DML training effectively in general.

We do another experiment with ResNet152 on both B4 and Internet2 network topologies, which gradually increase the lower bound up to the upper bound of the available bandwidth range (i.e., from 500 Mbps to 5 Gbps). Generally, we can observe that with a higher minimum bandwidth, we can achieve a lower communication time. However, this may not always be the case in this experiment, since we changed the bandwidths of all links in a reasonable range while increasing the minimum available bandwidth to avoid the fact that the APSP-selected PS node may not change with the only slight change of the worst link. Hence, the increasing of communication time is possible, because the available bandwidths of other



(a) Communication time per epoch over B4

(b) Communication time per epoch over Internet2

Fig. 4. Communication time per epoch comparing with node selecting methods.

TABLE III
NUMERICAL RESULT OF COMMUNICATION TIME COMPARING WITH NODE SELECTING METHODS.

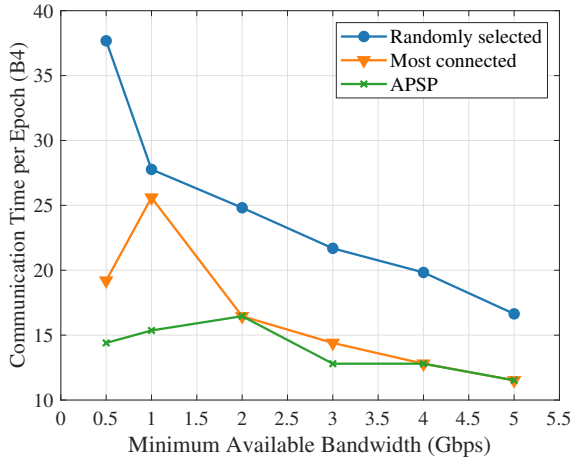
Topology	Training CNN	Communication time (s)		
		Randomly selected PS	Most connected PS	APSP selected PS
B4	ResNet50	14.88	7.36	**5.88**
	ResNet152	37.68	19.20	**14.40**
	AlexNet	39.74	20.48	**18.20**
	VGG-Net	87.18	44.16	**39.26**
Internet2	ResNet50	12.54	12.26	**9.82**
	ResNet152	32.72	32.00	**25.60**
	AlexNet	34.90	34.14	**27.30**
	VGG-Net	75.24	73.60	**58.88**

links may be decreased though the minimum one is increased. As depicted in Fig. 5, APSP performs better than other two solutions throughout the experiment. Moreover, as the lower bound of available bandwidth increases, the acceleration effect gradually becomes less significant. In particular, when the lower bound is raised to upper bound, APSP and selecting the most connected PS have the same performance with a high probability, since all the inter data center links maintain the same available bandwidth in this case. However, when the link bandwidths differ greatly, in another word, the lower bound remains large gap to upper bound, the solution that selects the most connected PS might performs even worse than random selection, depending on the topology. Therefore, APSP is much suitable and scalable for Geo-DML over heterogenous WAN.

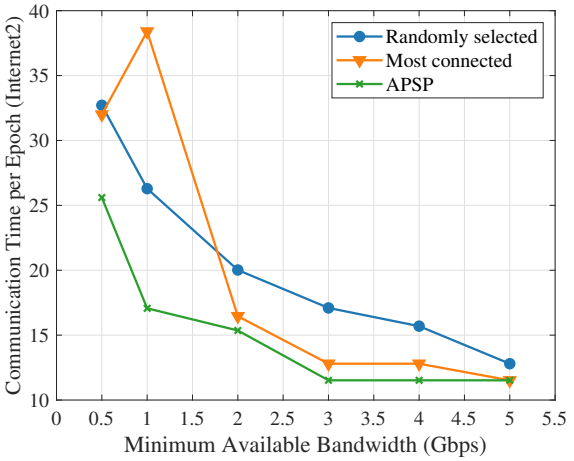
To demonstrate the routing performance of APSP, we compare it with two commonly used routing methods, which are shortest path routing and load balancing routing. The setup is same as the experiment in the previous paragraph, while the PS node is fixed, already selected by APSP. In another words, shortest path routing and load balancing routing are used to determine the path between worker nodes and the fixed PS node. In Fig. 6, APSP still has the lower communication time compared to the other routing methods, though the PS locations are same. When using shortest path routing, each flow follows its own shortest path. Thus, the lower weighted links are used more often and become the bottleneck of the network, which slow down the transmission. Load balancing routing focuses on balancing the load over the network. Hence, a flow may follow a path with more hops to spread the pressure causing longer communication time. Running APSP can determine the routing path by solving the LP problem, whose optimization objective is overall communication time. Therefore, our proposed algorithm can not only select the proper PS location, but also have better routing performance than the common routing methods.

VI. CONCLUSION

Our work is motivated by the widespread use of Geo-DML and the PS architecture over heterogeneous WAN. In this paper, we studied the PS placement problem in Geo-DML system to reduce the high communication cost during training. We first formulated the problem as an optimization model and proved it to be NP-hard. Because of the complexity of the problem, we proposed an approximation algorithm named APSP utilizing the method of randomized rounding to solve the problem efficiently. Thereafter, we analyzed the approximation performance of the APSP algorithm and gave the approximation factor for link capacity constraints. Extensive

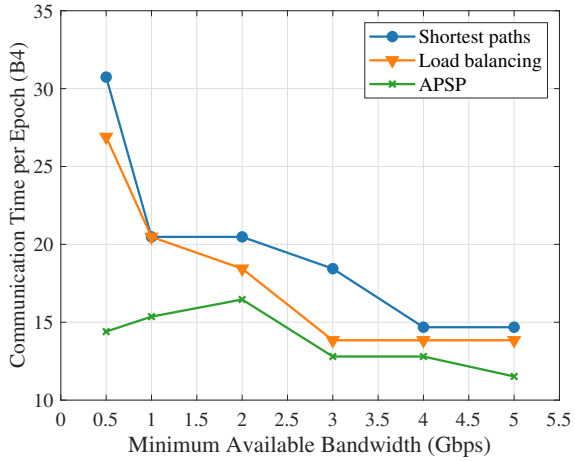


(a) Communication time vs. minimum available bandwidth (B4)

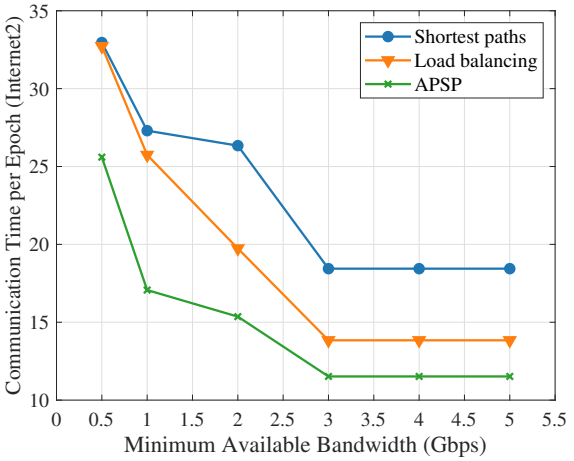


(b) Communication time vs. minimum available bandwidth (Internet2)

Fig. 5. Communication time per epoch vs. minimum available bandwidth comparing with node selecting methods.



(a) Communication time per epoch over B4



(b) Communication time per epoch over Internet2

Fig. 6. Communication time per epoch vs. minimum available bandwidth comparing with routing methods.

simulation results showed that the proposed algorithm had good approximation performance and significantly reduced the communication cost compared with the other two solutions, especially in the situation that link bandwidths vary greatly. However, the communication time will still slow down the training in PS architecture, since all the worker data centers have to send the large amount of data to a single central PS via WAN with scarce bandwidth resources. Therefore, job scheduling problem in Geo-DML on decentralized architecture will be considered in our future research.

REFERENCES

[1] X. Wu *et al.*, "Top 10 algorithms in data mining," *Knowledge inf. syst.*, vol. 14, no. 1, pp. 1–37, Jan. 2008.

[2] A. Frome *et al.*, "DeViSE: A deep visual-semantic embedding model," in *Proc. NIPS*, 2013.

[3] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[4] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, Aug. 2018.

[5] Google datacenter locations. http://www.google.com/about/datacenters/inside/locations/.

[6] Microsoft datacenters. http://www.microsoft.com/en-us/server-cloud/cloud-os/global-datacenters.aspx.

[7] AWS. http://aws.amazon.com/about-aws/global-infrastructure/.

[8] L. A. Bygrave, "Data privacy law: An international perspective," *Oxford: Oxford University Press*, vol. 10, 2014.

[9] J. M. Fromholz, "The european union data privacy directive," *Berk. Tech. LJ*, vol. 15, p. 461, 2000.

[10] K. Hsieh *et al.*, "Gaia: Geo-distributed machine learning approaching LAN speeds," in *Proc. USENIX NSDI*, 2017.

[11] S. Akter and M. Adnan, "WeightGrad: Geo-distributed data analysis using quantization for faster convergence and better accuracy," in *Proc. ACM KDD*, 2020.

[12] A. Zhou *et al.*, "Privacy regulation aware process mapping in geo-distributed cloud data centers," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 8, pp. 1872–1888, Aug. 2019.

[13] P. Raghavan and C. D. Tompson, "Randomized rounding: A technique for provably good algorithms and algorithmic proofs," *Combinatorica*, vol. 7, pp. 365–374, Dec. 1987.

[14] A. Ahmed, M. Aly, J. Gonzalez, S. M. Narayanamurthy, and A. J. Smola,

"Scalable inference in latent variable models," in *Proc. ACM WSDM*, 2012.
[15] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," *arXiv abs/1603.04467*, 2016.
[16] Q. Pu *et al.*, "Low latency geo-distributed data analytics," in *ACM SIGCOMM Comput. Comm. Review*, 2015.
[17] A. Vulimiri, C. Curino, B. Godfrey, K. Karanasos, and G. Varghese, "WANalytics: Geo-distributed analytics for a data intensive world," in *Proc. ACM SIGMOD*, 2015.
[18] A. Vulimiri *et al.*, "Global analytics in the face of bandwidth and regulatory constraints," in *Proc. USENIX NSDI*, 2015.
[19] K. Kloudas, R. Rodrigues, N. M. Preguiça, and M. Mamede, "Pixida: Optimizing data parallel jobs in wide-area data analytics," in *Proc. ACM VLDB*, 2015.
[20] C. C. Hung, L. Golubchik, and M. Yu, "Scheduling jobs across geo-distributed datacenters," in *Proc. ACM SoCC*, 2015.
[21] R. Viswanathan, G. Ananthanarayanan, and A. Akella, "CLARINET: WAN-aware optimization for analytics queries," in *Proc. USENIX OSDI*, 2016.
[22] I. Cano, M. Weimer, D. Mahajan, C. Curino, and G. M. Fumarola, "Towards geo-distributed machine learning," in *Proc. ACM CoRR*, 2016.
[23] Y. Zhao *et al.*, "Rapier: Integrating routing and scheduling for coflow-aware data center networks," in *Proc. IEEE INFOCOM*, 2015.
[24] M. Chowdhury, M. Zaharia, J. Ma, M. I. Jordan, and I. Stoica, "Managing data transfers in computer clusters with orchestra," in *Proc. ACM SIGCOMM*, 2011.
[25] M. Chowdhury, Y. Zhong, and I. Stoica, "Efficient coflow scheduling with varys," in *Proc. ACM SIGCOMM*, 2014.
[26] F. R. Dogar, T. Karagiannis, H. Ballani, and A. Rowstron, "Decentralized task-aware scheduling for data center networks," in *Proc. ACM SIGCOMM*, 2014.
[27] C. Hung, L. Golubchik, and M. Yu, "Scheduling jobs across geo-distributed datacenters," in *Proc. ACM SoCC*, 2015.
[28] L. Liu *et al.*, "Online job scheduling for distributed machine learning in optical circuit switch networks," *Knowledge-Based Syst.*, vol. 201–202, p. 106002, Aug. 2020.
[29] S. Even, A. Itai, and A. Shamir, "On the complexity of time table and multi-commodity flow problems," *Proc. IEEE SFCS*, 1975.
[30] S. Jain *et al.*, "B4: Experience with a globally-deployed software defined WAN," *ACM SIGCOMM Comput. Comm. Review*, vol. 43, no. 4, pp. 3–14, Aug. 2013.
[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE CVPR*, 2016.
[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012.
[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

APPENDIX A
PROBABILITY BOUND ANALYSIS

We analyze the probability bound of violating link bandwidth constraints after rounding as follows:

Theorem 3. *Our algorithm achieves the approximation factor of* $(2\ln n + 2\sqrt{\ln^2 n + 2\alpha \ln n})/\alpha + 1$ *after randomized rounding for link bandwidth constraints.*

Proof. Our algorithm solves *Problem 4* and get variables z_{fp} and m from which we can calculate the $\hat{x}_{fp}$. Notice that $\hat{x}_{fp}$ is fractional which means the data flow can be separated and transmitted by different path. Then we use randomized rounding method to determine the only one feasible path for each flow. We use $d_f(e)$ to denote the bandwidth occupied by flow $f \in F$ on link e. For each link $e \in E$ and each flow $f \in F$, we define $d_f(e)$ as:

$$d_f(e) = \begin{cases} \delta_{efp} \cdot |p| \cdot D, & with\ probability\ of\ \hat{x}_{fp} \\ 0, & otherwise. \end{cases} \tag{11}$$

For convenience, we use $h_{fp}(e)$ to replace $\delta_{efp} \cdot |p| \cdot D$. It is obvious that $d_{f1}(e), d_{f2}(e), \cdots$ are mutually independent. Thus, we can derive the expectation of occupied bandwidth on link e as:

$$\begin{aligned} E\left[\sum_{f\in F} d_f(e)\right] &= \sum_{f\in F} E\left[d_f(e)\right] \\ &= \sum_{f\in F}\sum_{p\in P_f} \hat{x}_{fp}\delta_{efp} \cdot |p| \cdot D \\ &= \sum_{f\in F}\sum_{p\in P_f} \hat{x}_{fp} h_{fp}(e) \\ &\le B_e. \end{aligned} \tag{12}$$

We define a new variable α:

$$\alpha = \min\{\frac{B_e}{h_{fp}(e)}, \forall e \in E, \forall f \in F, \forall p \in P_f\} \tag{13}$$

Combining (12) with the definition of α, we can derive:

$$\begin{cases} \dfrac{\alpha \cdot d_{fp}(e)}{B_e} \in [0,1] \\ \mathbb{E}\left[\displaystyle\sum_{f\in F} \frac{\alpha \cdot d_{fp}(e)}{B_e}\right] \le \alpha. \end{cases} \tag{14}$$

Here we introduce two lemmas to help analysis.

Lemma 1. (Chernoff bound) *There are* n *independent real variables* $x_1, x_2, \cdots, x_n$, *where* $x_i \in [0,1]$. *With* $\mu = E\left(\sum_{i=1}^{n} x_i\right)$, *there is:*

$$\Pr\left[\sum_{i=1}^{n} x_i \ge (1+\rho)\mu\right] \le e^{\frac{-\rho^2\mu}{2+\rho}}, \tag{15}$$

where ρ *is an arbitrary positive number.*

Lemma 2. (Union bound) *There are* n *events* $\Psi_1, \Psi_2, \cdots, \Psi_n$, *and each event happens with probability* $\Pr(\Psi_i)$. *Then,*

$$\Pr[\Psi_1 \cup \Psi_2 \cup \cdots \cup \Psi_n] \le \sum_{i=1}^{n} \Pr[\Psi_i]. \tag{16}$$

Applying Lemma 1, we can get:

$$\begin{aligned} \Pr\left[\sum_{f\in F} \frac{\alpha \cdot d_{fp}(e)}{B_e} \ge (1+\rho)\alpha\right] &\le e^{\frac{-\rho^2\alpha}{2+\rho}} \\ \Pr\left[\sum_{f\in F} \frac{d_{fp}(e)}{B_e} \ge 1+\rho\right] &\le e^{\frac{-\rho^2\alpha}{2+\rho}}, \end{aligned} \tag{17}$$

where ρ is an arbitrary positive number.

Next, we assume that:

$$\Pr\left[\sum_{f\in F} \frac{d_{fp}(e)}{B_e} \ge 1+\rho\right] \le e^{\frac{-\rho^2\alpha}{2+\rho}} \le \frac{\varphi}{n^2}, \tag{18}$$

where n is a variable related to the network structure, e.g., the number of nodes and φ is a function of n, when $n \to \infty$,

$\varphi \to 0$. Here we simply set $\varphi = 1/n^2$. Then we can rewrite (18) as:

$$\Pr\left[\sum_{f\in F}\frac{d_{fp}(e)}{B_e} \geq 1+\rho\right] \leq e^{\frac{-\rho^2\alpha}{2+\rho}} \leq \frac{1}{n^4} \tag{19}$$

From (19) we can solve $\rho \geq (2\ln n + 2\sqrt{\ln^2 n + 2\alpha \ln n})/\alpha$.

Applying Lemma 2, we can derive that:

$$\begin{aligned}
&\Pr\left[\bigcup_{e\in E}\left(\sum_{f\in F}\frac{d_{fp}(e)}{B_e} \geq 1+\rho\right)\right] \\
&\leq \sum_{e\in E}\Pr\left[\sum_{f\in F}\frac{d_{fp}(e)}{B_e} \geq 1+\rho\right] \\
&\leq |E|\,\frac{1}{n^4} \\
&\leq \frac{1}{n^2},
\end{aligned} \tag{20}$$

where $\rho \geq (2\ln n + 2\sqrt{\ln^2 n + 2\alpha \ln n})/\alpha$ and $|E|$ denote the link number.

Therefore, The link bandwidth constraints will not be violated after randomized rounding by a factor of $\rho + 1 = (2\ln n + 2\sqrt{\ln^2 n + 2\alpha \ln n})/\alpha + 1$ for the solution of *Problem 4*. □

Yongyao Li received the B.S. degree from University of Electronic Science and Technology of China, Chengdu, China, in 2015, the M.S. degree from Shijiazhuang Railway Institute, Shijiazhuang, China, in 2002. He is currently pursuing the Ph.D degree in Institute of Systems Engineering at Macau University of Science and Technology. His interests include Intelligent System Theory and Application.

Chenyu Fan is currently pursuing the Master degree in Information and Communication Engineering with the School of Communication and Information Engineering, University of Electronic Science and Technology of China, Chengdu, China. Her research interests include network optimization, edge computing, and artificial intelligence.

Xiaoning Zhang received the B.S., M.S., and Ph.D. degrees in Communication and Information Engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2002, 2005, and 2007, respectively. He is currently an Professor with School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, China. His research interests include network design, software defined networks and network function virtualization.

Yufeng Chen received the B.S. and Ph.D. degrees from Xidian University, Xi'an, China, in 2006 and 2011, respectively. He is the author or coauthor of over 40 publications. He is a coauthor with ZhiWu Li, Optimal Supervisory Control of Automated Manufacturing Systems (CRC Press, Taylor & Francis Group, 2013). He is with the Institute of Systems Engineering, Macau University of Science and Technology, Macau, China. His research interests include Petri net theory and applications, supervisory control of discrete event systems, and scheduling flexible manufacturing systems.