2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)

Towards Maximal Service Profit in Geo-Distributed Clouds

Zhenjie Yang[1], Yong Cui[1]*, Xin Wang[2], Yadong liu[1], Minming Li[3,4] and Zhixing Zhang[1]
[1]*Department of Computer Science and Technology, Tsinghua University, Beijing, China*
[2]*Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, New York, USA*
[3]*Department of Computer Science, City University of Hong Kong, Hong Kong, China*
[4]*City University of Hong Kong Shenzhen Research Institute, Shenzhen, China*
yangzj15@mails.tsinghua.edu.cn, cuiyong@tsinghua.edu.cn, x.wang@stonybrook.edu, minming.li@cityu.edu.hk

***Abstract*—With the proliferation of globally-distributed services and the quick growth of user requests for inter-datacenter bandwidth, cloud providers have to lease a good deal of bandwidth from Internet service providers to satisfy the user demands. Neither maximizing the service revenue nor minimizing the service cost can bring the maximal service profit to cloud providers. The diversity of user requests and the large unit of inter-datacenter bandwidth further increase the difficulty of scheduling user requests. In this paper, we propose a cloud operational model to help cloud providers to make more service profit by properly selecting requests to serve rather than serving all user requests. We formulate the problem of service profit maximization and prove its NP-hardness. Considering the complicated coupling between maximizing revenue and minimizing cost, we propose a framework, Metis, for the efficient scheduling of user requests over inter-datacenter networks to maximize the service profit for cloud providers. Metis is formed with the alternate operations of two algorithms derived from randomized rounding techniques and Chernoff-Hoeffding bound. We prove that they can provide the guarantees on approximation ratios. Our extensive evaluations demonstrate that Metis can achieve more than 1.3x the service profits of existing solutions.**

***Keywords*-geo-distributed cloud; service profit; maximization**

I. Introduction

In the past decade, cloud computing has made great success and transformed a large part of the IT industry [1]. As many companies move their services to clouds, cloud providers, such as Amazon, Microsoft and Alibaba, have made large profits from the cloud computing market. With the proliferation of globally-distributed services and quick growth of user requests across data centers, cloud providers turn to geo-distributed clouds [2], [3].

Cloud providers build data centers in different geographical areas and use inter-datacenter wide area networks (Inter-DC WANs) to connect them [3]–[6]. Inter-DC WAN is a dedicated network managed by a cloud provider [2]. Except some giant companies that build dedicated lines between their data centers, most cloud providers lease bandwidth from Internet service providers (ISPs) to connect their data centers, with the cost up to hundreds of millions of dollars per year. Typically ISPs sell bandwidth at a fixed price per *unit*, e.g., 10Gbps or 80Gbps [2], [7], [8]. The price varies with links and regions [9]. The bandwidth usage is calculated over a fixed *billing cycle*, such as a month or a year [8]. As the growth of WAN bandwidth has been decelerating for many years and the volume of Inter-DC data increases quickly [10], [11], WAN bandwidth becomes more and more expensive and it is critical for cloud providers to use the bandwidth more economically in order to make higher profit. On the cloud computing market, customers expect cloud providers to reserve a certain amount of exclusive bandwidth between given data centers in specific periods for them to use and are willing to pay a certain amount of fee to cloud providers as return. To satisfy the customer requests, cloud providers may have to purchase bandwidth from ISPs at high cost. It is important and challenging for a cloud provider to maximize its service profit, the net benefit that is equal to the service revenue minus the bandwidth cost.

Current service mode on the cloud computing market prevents cloud providers from making the maximum service profit due to the high cost of Inter-DC WANs, as providers accept *all* customer requests without considering the bandwidth cost. Inspired by the successful experience of *the first-price sealed-bid auction* in auction market, where bidders bid simultaneously and the one with the highest bid will receive the item [12], we consider a cloud operational model where customers submit their transfer requirements and bid to the cloud provider independently and simultaneously, and the cloud provider evaluates these requests and accepts the ones that can maximize its service profit.

Maximizing service profit requires cloud providers to address two major challenges: (1) *Request acceptance.* Although we have shown that accepting all the requests of customers may not contribute to the maximal service profit, it is still hard for cloud providers to find the right set of requests to turn down. If there are N $(N \geq 0)$ requests, there will be 2^N combinations of requests for cloud providers to choose from. As there is no function to describe the service profit changes over different combinations, it would be very time-consuming to evaluate these combinations one by one. (2) *Request scheduling.* In Inter-DC WANs, there are several routing paths between two data centers. For a

* Yong Cui is the corresponding author.

2575-8411/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDCS.2019.00051

set of accepted requests, adopting different routing solutions will lead to different bandwidth costs and greatly affect the service profit. Properly scheduling these requests is essential but it is difficult for cloud providers to find the right schedule and path for the maximum service profit.

Intensive efforts have been made to schedule requests between data centers economically [8], [13]–[17]. Many solutions make full use of the purchased bandwidth with store-and-forward approaches or multipath transmission, which introduces costs for extra storage or packet-level reordering. Some other efforts are made to schedule Inter-DC transfers with different pricing methods (e.g., dynamic pricing) to maximize the service revenue, which requires the cloud providers to modify the current pricing mechanism. Overall, these solutions are not good options to apply in practical geo-distributed clouds.

As the above challenges are intertwined with each other, a possible way to maximize the service profit is the alternate optimization of the service revenue under given bandwidth and the minimization of bandwidth cost under given requests, based on which cloud providers could dynamically adjust the bandwidth to purchase and the requests to accept. In this paper, we propose an easy-to-control alternate mechanism and take one step further to deal with the problem of service profit maximization (SPM). We design a framework, Metis, to enable cloud providers to maximize their service profits by alternately solving two variants of SPM, request-limited SPM (RL-SPM) and bandwidth-limited SPM (BL-SPM), both of which are solved with guaranteed approximation ratios. The key contributions are summarized as follows:

- To the best of our knowledge, this is the *first* work that formulates the problem of SPM in geo-distributed clouds, in which we take into account the variation of bandwidth prices in Inter-DC WANs. We show the problem is NP-hard by reducing the subset sum problem to our problem.
- Benefitting from the alternate optimization of RL-SPM and BL-SPM, Metis solves the SPM problem in polynomial time without incurring too high loss. It also enables cloud providers to balance the profit performance and time complexity by tuning the parameters of Metis.
- Based on randomized rounding techniques, we design an algorithm MAA to solve RL-SPM and propose a technique to prove its approximation ratio by combining two other approximation ratios. We also develop a deterministic method TAA to solve BL-SPM with a guaranteed approximation ratio based on the Chernoff-Hoeffding bounds. Extensive evaluations demonstrate that Metis can achieve more than 130% the service profit of the existing solutions.

The remainder of this paper is organized as follows: we introduce SPM and Metis in Section II. We present two algorithms, MAA and TAA in Section III and Section IV. We show the evaluation results in Section V, the related work in Section VI and the conclusion in Section VII.

II. Service Profit Maximization

In this section, we first introduce the basic system model and notations, then we formulate the problem of service profit maximization and prove its NP-hardness. We also define two variants of it to facilitate the design of our framework, Metis.

A. Model

A cloud provider controls an Inter-DC network $G(V, E)$. Each edge e in G represents a directed link from a data center to another. We use V and E to denote the set of data centers and edges in the network G. ISPs charge for bandwidth in a fixed billing cycle [8]. A billing cycle consists of T independent time slots, and we use t to index a time slot. With u_e denoting the cost of per unit of bandwidth on edge e and c_e the bandwidth to be charged, the bandwidth cost of e is the product of the two [14], [18]. Typically, c_e is an integer in practice [8]. In a billing cycle, intra-cloud services send a set of K transfer requests. A request i ($1 \leq i \leq K$) is denoted by a six-tuple $\{s_i, d_i, ts_i, td_i, r_i, v_i\}$, where the transmission is carried from the source data center s_i to the destination data center d_i between the time ts_i and td_i. The bandwidth required is r_i, and the value is v_i. For each pair of data centers (s_i, d_i), there are multiple available paths in G. We use $P_i = \{P_{i,j} : 1 \leq j \leq L_i\}$ to denote the set of paths for request i, where L_i denotes the number of paths. With the above definitions, we define service revenue and service cost as follows.

Service Revenue. For a request i, we use a binary variable $x_{i,j}$ to indicate whether it flows through path $P_{i,j}$, where $x_{i,j}$ equals 1 if yes, and 0 otherwise. If its requirement is satisfied, i.e., $\sum_{j=1}^{L_i} x_{i,j} = 1$, the cloud provider receives a service revenue of v_i, and the revenue is 0 otherwise. The service revenue $\mathcal{I}$ can be represented as:

$$\mathcal{I} = \sum_{i=1}^{K} v_i (\sum_{j=1}^{L_i} x_{i,j})$$

Service Cost. In this work, we focus on the cost of Inter-DC WANs and take the bandwidth cost of Inter-DC links as service cost, and

$$\mathcal{C} = \sum_{e \in E} u_e c_e.$$

For clarity, we summarize the notations we use in Table I.

B. Formulation

For any pair of data centers in the network G, there is at least one routing path. For request i, the cloud provider should select at most one path to reserve bandwidth. Thus the following inequality should be satisfied:

$$\forall i \quad : \quad \sum_{j=1}^{L_i} x_{i,j} \leq 1 \tag{1}$$

Table I. Summary of notations

Notation	Definition
G	(V, E): the topology of Inter-DC WAN
K	the number of requests in a billing cycle
T	the number of time slots of a billing cycle
i	$\{s_i, d_i, ts_i, td_i, r_i, v_i\}$: the i-th request; s_i: source; d_i: destination; ts_i: start time; td_i: end time; r_i: required rate; v_i: value;
L_i	the number of directed path from s_i to t_i
$P_{i,j}$	$\{i, j \mid 1 \leq i \leq K, 1 \leq j \leq L_i\}$: the j-th directed path from s_i to t_i
P_i	$\{P_{i,j} : 1 \leq j \leq L_i\}$: the set of directed paths from s_i to t_i
u_e	bandwidth price of e, i.e., the cost of per unit of bandwidth on e
c_e	the charging bandwidth of e
$x_{i,j}$	$\{0, 1\}$: whether request i flows through $P_{i,j}$
$I_{i,j,e}$	$\{0, 1\}$: whether edge e is in the path $P_{i,j}$
$\mathcal{I}$	service revenue
$\mathcal{C}$	service cost

Our goal is to find the match of requests and transmission paths, which we denote as "schedule". A schedule is said to satisfy the request i if $\sum_{j=1}^{L_i} x_{i,j} = 1$ holds.

A schedule is said to satisfy the link capacities if the link capacity constraint is satisfied for every edge e at any time slot t. First, we describe the required bandwidth r_i of request i as:

$$\forall i, \forall t \quad : \quad r_{i,t} = \begin{cases} r_i \ , & t \in [ts_i, td_i] \\ 0 \ , & \text{otherwise} \end{cases}$$

Based on this, we present the link capacity constraint as:

$$\forall t, \forall e \quad : \quad \sum_{i=1}^{K} \sum_{j=1}^{L_i} r_{i,t} x_{i,j} I_{i,j,e} \leq c_e \tag{2}$$

where $I_{i,j,e}$ is a predefined binary number and it indicates whether edge e is on path $P_{i,j}$.

To maximize service profit for a cloud provider, our objective is to maximize the value of service revenue minus service cost in Inter-DC WANs. We call it service profit maximization (SPM), and formulate it as follows:

$$\max \ (\mathcal{I} - \mathcal{C})$$

s.t.

$$\forall e \quad : \quad c_e \in \mathbb{N} \tag{3}$$

$$\forall i, \forall j \quad : \quad x_{i,j} \in \{0, 1\} \tag{4}$$

$$\text{Constraint } (1)(2)$$

By reducing the subset sum problem [19] (denoted as SUBSET-SUM) to SPM, we have the following theorem:

Theorem 1. *SPM is NP-hard.*

Proof: SPM is NP-hard, as it contains SUBSET-SUM as a special case, which is NP-complete. SUBSET-SUM is defined as follows: Given a set of integers $\mathcal{S}$, is there a non-empty subset whose sum is a given integer $\mathcal{N}$?

We first construct an instance $\mathcal{A}$ of SUBSET-SUM. We consider setting the integer set $\mathcal{S} = \{a_1, a_2, ..., a_n\}$ and $\sum_{i=1}^{n} a_i = \mathcal{M}$. The goal of SUBSET-SUM is to find a subset of sum $\mathcal{N}$ in $\mathcal{S}$. Without loss of generality, let $\mathcal{N} < \mathcal{M} < 2\mathcal{N}$. Next, we construct a special case $\mathcal{A}'$ of SPM. Suppose there is one edge e in the network G and one time slot in a scheduling period, and there are n requests. The value of i is linearly related to the bandwidth it requires. For each request i, it requires r_i $(= a_i/\mathcal{N})$ units of bandwidth and we set the value v_i of it as r_i. Then we have that the sum of v_i is greater than 1 but less than 2. We consider setting the bandwidth price of edge e as $1 - \sigma$ where σ is less than 1 but it is infinitely close to 1. By far we have transformed $\mathcal{A}$ to $\mathcal{A}'$ in polynomial time.

For the instance $\mathcal{A}'$, the cloud provider would obtain the maximal service profit $1 - \sigma$ by satisfying a subset of sum 1 in the set of $\{v_i\}$. If we could solve SPM with a polynomial-time algorithm, we would obtain the optimal solution of SUBSET-SUM, which means that SUBSET-SUM could be solved in polynomial time. Therefore, SPM is at least as hard as SUBSET-SUM. As SUBSET-SUM is NP-hard, SPM is NP-hard, too. This completes the proof. ■

Despite that directly solving SPM is complex and diffcult because of the tight coupling of the service revenue and the service cost, solving its variants can be relatively easy, thus we define two variants of SPM as follows:

- **Variant #1:** Given a set of accepted requests, maximizing the service profit is equivalent to minimizing the service cost, which is called request-limited SPM (RL-SPM).
- **Variant #2:** Suppose the bandwidth in the network is fixed, service profit maximization can be transformed to achieving the maximal service revenue, which is called bandwidth-limited SPM (BL-SPM).

By alternately solving the two variants of SPM, we will efficiently solve it in the following.

C. Metis

Despite that the two variants of SPM are relatively easier to solve, solving either of the two is not sufficient for solving SPM. Considering the conflicts between RL-SPM and BL-SPM, where the given condition of one problem (i.e., the requests accepted in RL-SPM and the bandwidth limitation in BL-SPM) is the objective of the other (i.e., minimizing the bandwidth cost and maximizing the service revenue). By dynamically setting the accepted requests and the bandwidth limitation as required by the two variants, alternately solving them can potentially generate a good solution of SPM. Following this strategy, we propose a framework, **Metis**,

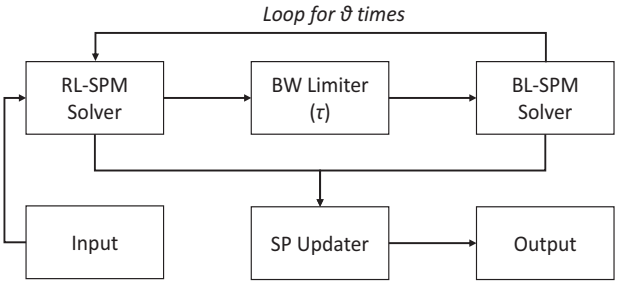


Figure 1. Framework of Metis

for cloud providers to specify the alternate mode, as shown in Fig. 1.

There are six modules in Metis: Input collects the necessary information for solving SPM, such as the requests, the network topology, the bandwidth prices and so on. Output will output the final decisions, i.e., acceptance decision and scheduling decision, based on the calculation of SP Updater. SP Updater works with other three modules: RL-SPM Solver, BW Limiter and BL-SPM Solver. Before running them, SP Updater initializes the service profit as zero, where the cloud provider accepts no request and uses no bandwidth. Both service revenue and service cost are zero. We introduce the functions and operations of these modules in the following.

- **RL-SPM Solver.** Given a set of accepted requests, it aims to present a routing solution that needs as little bandwidth as possible. We run it to output the bandwidth to be charged and the scheduling of requests, then we calculate the service profit. If it is greater than the recorded service profit by SP Updater, SP Updater updates the service profit as the current one and record the current schedule. In the initialization phase, we set all the user requests as "accepted".
- **BW Limiter.** Before running the BL-SPM Solver, it is used to limit the bandwidth with a rule predetermined by the cloud provider and denoted as τ. In this paper, we set τ as the rule that BW Limiter uses to reduce the bandwidth of link whose average utilization is the minimum. Based on the schedule obtained from RL-SPM Solver, BW Limiter sets the link bandwidth and calls BL-SPM Solver to maximize the service revenue.
- **BL-SPM Solver.** Under the limit of bandwidth obtained from BW Limiter, BL-SPM Solver accepts part of user requests and schedules them with the transmission paths to achieve the maximal service revenue. In the case that the given bandwidth can not satisfy all requests, the BL-SPM Solver will decline some requests and update the set of user requests.
- **SP Updater.** It is in charge of the update and the record of service profit when RL-SPM Solver or BL-SPM Solver outputs a better schedule. It also records the acceptance decision and bandwidth allocation.

In a scheduling period, the above modules will run for θ ($\theta \geq 1$) times before Metis outputs the final decisions. Through Metis, we alternately reduce the cost and improve the revenue to optimize the service profit. Cloud providers can set τ and θ based on their actual needs (e.g., low computing time) and historical data [6], [20]. Metis has several advantages: (1) Easy-to-control. Cloud providers can tune it freely by setting parameters. (2) Convergence. As BL-SPM Solver sometimes declines user requests, the number of user requests to be scheduled decreases with the increasing number of completed loops. Metis loops at most K (the number of requests in a scheduling period) times before finishing scheduling all requests.

In Metis, the efficiency of RL-SPM Solver and BL-SPM Solver greatly affects the final service profit. In the following sections, we design algorithms to solve RL-SPM and BL-SPM with guaranteed approximation ratios.

III. Multistage Approximation Algorithm

RL-SPM Solver is applied to select routing paths for the requests and minimize the total bandwidth cost. However, there are two challenges: a request is unsplittable and the bandwidth of links in Inter-DC WANs is integer, which make it hard to solve the problem directly.

To address the challenges, we propose to decompose RL-SPM into two subproblems: a subproblem $\mathcal{P}_1$ with the relaxing of the integer requirement of bandwidth, and a subproblem $\mathcal{P}_2$ with the relaxing of the integral routing constraint of request. $\mathcal{P}_1$ becomes the well-known multicommodity unsplittable flow problem [19]. Then we design a multistage approximation algorithm (**MAA**) to independently solve $\mathcal{P}_1$ and $\mathcal{P}_2$ based on the relax-and-round method, as follows:

- **Relaxation:** We first solve the relaxed linear program of $\mathcal{P}_1$ by allowing $x_{i,j}$ to be a fractional number, with $x_{i,j} \in [0,1]$. An advanced LP solver can yield the optimal fractional results, denoted as $\{\hat{x}_{i,j}\}$ and $\{\hat{c}_e\}$.
- **Randomized rounding:** With the randomized rounding scheme, a request i is selected to transmit over the path $P_{i,j}$ with a probability $\hat{x}_{i,j}$. For each request i, the overall path selection factors over all links should meet $\sum_{j=1}^{L_i} \hat{x}_{i,j} = 1$. We select exactly one path for each request i with a probability $\hat{x}_{i,j}$. Let $|E|$ denote the number of edges in the network G, it achieves a $O(\frac{\log|E|}{\log\log|E|})$-approximation solution for $\mathcal{P}_1$ with a high probability [21], [22].
- **Ceiling:** As the start times and end times of requests are different, for cloud providers, the required bandwidth varies with time slots and edges. Considering that bandwidth is charged in integer units in practice, for each edge e, we round up $\hat{c}_e$ as the bandwidth for charging and denote it as $\lceil \hat{c}_e \rceil$, and calculate the total bandwidth cost. We denote the set of edges whose fractional charging bandwidth $\hat{c}_e$ is positive as E' ($E' \subseteq E$).

Algorithm 1: MAA: Multistage Approximation Algorithm

Input: Requests: $\{1, 2, ..., K\}$; Paths: $\{P_1, P_2, ..., P_K\}$; Bandwidth prices: $\{u_e\}$;
Output: Transmission paths for requests: $\{x_{i,j}\}$; Charging bandwidth: $\{c_e\}$;

1 Relax RL-SPM and solve its relaxed linear program
2 **for** $1 \leq i \leq K$ **do**
3 | Select exactly one path $P_{i,j}$ with probability $x_{i,j}$
4 **end**
5 Initialize each c_e as 0
6 **for** $e \in E$ **do**
7 | $c_e = \lceil \max_t \{\sum_{i=1}^{K} r_{i,t} \hat{x}_{i,j} I_{i,j,e}\} \rceil$
8 **end**
9 **return** $\{x_{i,j}\}$ and $\{c_e\}$

In the case that any fractional bandwidth $\hat{c}_e$ is greater than a positive number α, i.e., $\alpha = \min_{e \in E'} \{\hat{c}_e\}$, we have a theorem about the approximation algorithm for $\mathcal{P}_2$. Before presenting it, we denote the ρ-approximation algorithm of a minimization problem as the one that can achieve a solution within ρ ($\rho \geq 1$) times the optimal solution. Similar to the definition of ρ-approximation algorithm, the ρ-relaxed algorithm for an integer programming minimization problem is the one that can achieve a solution within ρ times the optimal solution of the relaxed original problem. Based on our definitions, we have the following theorem:

Theorem 2. *There exists a $(\frac{\alpha+1}{\alpha})$-relaxed algorithm for $\mathcal{P}_2$, where $\alpha = \min_{e \in E'} \{\hat{c}_e\}$.*

Proof: With the paths selected for requests in randomized rounding procedure, we denote the minimum service cost of the relaxed $\mathcal{P}_2$ as $\ddot{\mathcal{C}}_2$ and the output objective of ceiling procedure as $\hat{\mathcal{C}}_2$. For each edge $e \in E'$, the bandwidth for charging is denoted by $\hat{c}_e$ and $\lceil \hat{c}_e \rceil$ respectively. Then we have:

$$\begin{aligned} \ddot{\mathcal{C}}_2 &= \sum_{e \in E'} u_e \hat{c}_e \\ \hat{\mathcal{C}}_2 &= \sum_{e \in E'} u_e \lceil \hat{c}_e \rceil \end{aligned}$$

According to the operation in ceiling procedure, it is evident that

$$\forall e \in E' : \ \hat{c}_e \leq \lceil \hat{c}_e \rceil < \hat{c}_e + 1$$

As $\hat{c}_e \geq \alpha$ holds for any edge e in E', it is easy to prove that

$$\forall e \in E' : \ \lceil \hat{c}_e \rceil < \frac{1+\alpha}{\alpha} \cdot \hat{c}_e$$

Then we have

$$\hat{\mathcal{C}}_2 < \frac{\alpha+1}{\alpha} \cdot \ddot{\mathcal{C}}_2$$

This completes the proof. ■

MAA is presented in **Algorithm 1**. On line 1, we solve the relaxation of $\mathcal{P}_1$. From line 2 to line 4, we select a path for each request according to the probability. On line 5, we initialize the bandwidth to be charged to zero. For each link $e \in E$, at each time slot t, we find the total bandwidth required by the requests that are scheduled to transmit through e and we round up the maximal fractional bandwidth to obtain the bandwidth for charging of e (from line 6 to line 8). On line 9, our algorithm returns the schedule and the charging bandwidth. We solve RL-SPM with MAA in polynomial time, which has a proved approximation ratio. The time complexity of MAA is $O(K \cdot T \cdot |E|)$, where K denotes the number of requests, T denotes the number of time slots in a billing cycle and $|E|$ denotes the number of edges in G. For MAA, we have the following theorem:

Theorem 3. *Suppose there exist a ρ_1-approximation algorithm ($\rho_1 \geq 1$) for $\mathcal{P}_1$ and a ρ_2-relaxed algorithm ($\rho_2 \geq 1$) for $\mathcal{P}_2$. Then there exists a $(\rho_1 \rho_2)$-approximation algorithm for RL-SPM.*

Proof: We denote RL-SPM as $\mathcal{P}_0$ and the optimal objective of $\mathcal{P}_0$ as $\mathcal{C}_0^*$. First, we use ρ_1-approximation algorithm to solve $\mathcal{P}_1$. Denoting the output objective as $\hat{\mathcal{C}}_1$ and $\mathcal{C}_1^*$ the optimal objective value of $\mathcal{P}_1$. With the selected paths $\hat{x}_{i,j}$ for each request i, we have

$$\hat{\mathcal{C}}_1 \leq \rho_1 \mathcal{C}_1^* \leq \rho_1 \mathcal{C}_0^*$$

Let $\ddot{\mathcal{C}}_2$ be the optimal solution of relaxed $\mathcal{P}_2$, then we have

$$\ddot{\mathcal{C}}_2 \leq \hat{\mathcal{C}}_1$$

Second, we use ρ_2-relaxed algorithm to solve $\mathcal{P}_2$, whose objective is denoted by $\hat{\mathcal{C}}_2$. Hence we have

$$\hat{\mathcal{C}}_2 \leq \rho_2 \ddot{\mathcal{C}}_2$$

Therefore, the following inequality

$$\hat{\mathcal{C}}_2 \leq \rho_2 \ddot{\mathcal{C}}_2 \leq \rho_2 \hat{\mathcal{C}}_1 \leq \rho_2 \rho_1 \mathcal{C}_1^* \leq \rho_2 \rho_1 \mathcal{C}_0^*$$

holds. This completes the proof. ■

Combining Theorem 2 and Theorem 3, we prove the approximation ratio of MAA in the following:

Theorem 4. *MAA is a $O(\frac{\alpha+1}{\alpha} \cdot \frac{\log |E|}{\log \log |E|})$-approximation algorithm for RL-SPM with a high probability.*

Proof: Let $\hat{\mathcal{C}}_0$ denote the total bandwidth cost of RL-SPM obtained with MAA and $\mathcal{C}_0^*$ denotes the minimal bandwidth cost. Our goal is to prove that

$$\hat{\mathcal{C}}_0 \leq O(\frac{\alpha+1}{\alpha} \cdot \frac{\log |E|}{\log \log |E|}) \cdot \mathcal{C}_0^* \tag{5}$$

Since randomized rounding procedure achieves a $O(\frac{\log |E|}{\log \log |E|})$-approximation ratio for $\mathcal{P}_1$ with high probability and ceiling procedure achieves a $(\frac{\alpha+1}{\alpha})$-approximation ratio for the relaxed $\mathcal{P}_2$, with Theorem 3,

we can prove that the above inequality (5) holds. This completes the proof. ■

IV. Tree-based Approximation Algorithm

Given the fixed bandwidth, the objective of BL-SPM Solver is to maximize the service revenue. Since BL-SPM contains unsplittable flow problem (USF) as a special case, which is known to be NP-hard [22], BL-SPM is also NP-hard. The proof is easy and omitted for brevity. The main challenge comes from that improper rounding will lead to the violation of link capacity constraint, which makes it hard to directly solve our problem. To address the issue, we propose a tree-based approximation algorithm (**TAA**) that exploits the decision tree to construct a feasible schedule without violating any link capacity constraint.

First, we relax the problem BL-SPM that $x_{i,j} \in [0,1]$ can be a fractional number to select the requests, and we denote the optimal scheduling solution of the relaxed problem as $\{\hat{x}_{i,j}\}$. The corresponding revenue is denoted by $\hat{\mathcal{I}}$. Similarly, we denote the optimal objective of BL-SPM as $\mathcal{I}^*$.

Since requests are independent from each other, we select a path $P_{i,j}$ for request i independently with a probability $x_{i,j}$ and denote by $\mathcal{I}_i$ the service revenue generated by request i. The expectation of $\mathcal{I}_i$ is:

$$E[\mathcal{I}_i] = v_i \sum_{j=1}^{L_i} x_{i,j} = m_i$$

where v_i denotes the value of request i. Then we have

$$E[\mathcal{I}] = \sum_{i=1}^{K} E[\mathcal{I}_i] = m$$

where K denotes the number of requests. Based on the following Chernoff-Hoeffding bounds [21], [23],

Theorem 5. ***(Chernoff-Hoeffding bounds)*** *Suppose $\mathcal{I}_1$, $\mathcal{I}_2$, ..., $\mathcal{I}_K$ are K independent random variables in $[0,1]$ and $\mathcal{I} = \sum_{i=1}^{K} \mathcal{I}_i$. For $\delta > 0$, and $m = E[\mathcal{I}] \geq 0$,*

$$Pr[\mathcal{I} > (1+\delta)m] < \left[\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right]^m$$

For $0 < \gamma \leq 1$,

$$Pr[\mathcal{I} < (1-\gamma)m] < \left[\frac{e^{\gamma}}{(1+\gamma)^{(1+\gamma)}}\right]^m$$

we normalize the expectation of $\mathcal{I}_i$ to $[0,1]$ to adapt to the above theorem, then we define $B(m,\delta)$ as:

$$B(m,\delta) = [\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}]^m$$

to be the probability that $\mathcal{I}$ is within a given δ ratio around m with the random selection of paths for requests. We also define $D(m,x)$ to replace δ and make the following equality hold:

$$B(m, D(m,x)) = x$$

Next, we use both $B(\cdot)$ and $D(\cdot)$ to guide the finding of feasible solutions of BL-SPM.

Since link capacity constraints may be violated if a path $P_{i,j}$ is selected for request i with the probability $\hat{x}_{i,j}$, we introduce a *scaling factor* μ ($0 < \mu < 1$), and take $\mu\hat{x}_{i,j}$ as the probability that we select the path $P_{i,j}$. To ensure that the probability of violating the link capacity constraint of BL-SPM is less than $\frac{1}{T\cdot(N+1)}$, we set μ to satisfy the following inequality:

$$B(\mu c, \frac{1-\mu}{\mu}) < \frac{1}{T\cdot(N+1)} \tag{6}$$

where c denotes the minimum bandwidth of edges in the WAN (except the edges whose bandwidth is zero); T and N denote the number of time slots in a billing cycle and the number of edges in G, respectively.

By our choice of μ, the probability that a constraint is violated will be less than $\frac{1}{T\cdot(N+1)}$. Since there are $T\cdot N$ link capacity constraint in BL-SPM, the probability that there exists at least one violated constraint is less than $\frac{N\cdot T}{T\cdot(N+1)}$. The expectation of the service revenue, $\mathcal{I}_S$, will be scaled down, i.e., $\mathcal{I}_S = \mu\hat{\mathcal{I}}$. Based on it, we have the following theorem:

Theorem 6. *Suppose $\mathcal{I}_B = \mathcal{I}_S \cdot [1 - D(\mathcal{I}_S, \frac{1}{N+1})]$, there exist solutions that make the service revenue of BL-SPM, $\widetilde{\mathcal{I}}$, satisfy:*

$$\widetilde{\mathcal{I}} \geq \mathcal{I}_B \tag{7}$$

Proof: If a solution makes the theorem hold, we say the corresponding schedule is "good", otherwise it is "bad". To prove the theorem, we will show that there exists non-zero probability that inequality (7) holds while no constraint in BL-SPM is violated (denoted as Y) in the following, i.e.,

$$Pr[\widetilde{\mathcal{I}} \geq \mathcal{I}_B \cap Y] > 0 \tag{8}$$

By inequality (7), we have

$$Pr[\widetilde{\mathcal{I}} < \mathcal{I}_B] < \frac{1}{N+1} \tag{9}$$

Since there are $N\cdot T$ link capacity constraints in BL-SPM, and the probability that any of them is violated is less than $\frac{1}{T\cdot(N+1)}$, the probability that there exists at least one violated link capacity constraint (denoted by $\bar{Y}$) is less than $\frac{N\cdot T}{T\cdot(N+1)}$. Then we have

$$\begin{aligned} & Pr[\widetilde{\mathcal{I}} < \mathcal{I}_B \cup \bar{Y}] \\ \leq\ & Pr[\widetilde{\mathcal{I}} < \mathcal{I}_B] + Pr[\bar{Y}] \\ <\ & \frac{1}{N+1} + T\cdot N \cdot \frac{1}{T\cdot(N+1)} = 1 \end{aligned} \tag{10}$$

Since the sum of $Pr[\widetilde{\mathcal{I}} < \mathcal{I}_B \cup \bar{Y}]$ and $Pr[\widetilde{\mathcal{I}} \geq \mathcal{I}_B \cap Y]$ equals 1 and $Pr[\widetilde{\mathcal{I}} < \mathcal{I}_B \cup \bar{Y}] < 1$, we have that $Pr[\widetilde{\mathcal{I}} \geq \mathcal{I}_B \cap Y] > 0$, thus inequality (8) holds, implying the solutions achieve objectives higher than $\mathcal{I}_B$. ■

As proven in Theorem 6, there exist good solutions for BL-SPM, we propose an algorithm to construct a good solution for BL-SPM based on conditional probabilities. We look for a good solution by means of a decision tree.

Consider a K-level tree $\mathcal{T}$ corresponding to K total requests, the nodes at level i represent the possible choices for request i. For any request i, there are $L_i + 1$ choices, either scheduling it on a path $P_{i,j}$ $(1 \leq j \leq L_i)$ or declining it. For the convenience of presentation of algorithm process, we consider declining request i as scheduling it on the path P_{i,L_i+1}, thus a request i has $L_i + 1$ available paths. For K requests, there are totally $\prod_{i=1}^{K}(L_i + 1)$ possible schedules, each represented as a leaf of the decision tree. Theorem 6 shows there exist good schedules for BL-SPM, i.e., good leaves in $\mathcal{T}$. We aim to walk down $\mathcal{T}$ from the root node to a good leaf in polynomial time so that the resulted service revenue $\widetilde{\mathcal{I}}$ is no less than $\mathcal{I}_B$.

At a typical stage of the computation, we are at some node at level i $(1 \leq i \leq K)$. It is evident that we have walked down the first $i-1$ levels. Now, we wish to select one of the $L_i + 1$ nodes to minimize the probability that we reach a leaf with bad schedule. We denote by $p_i(q_1, ..., q_{i-1})$ the conditional probability that we will reach a bad leaf given the paths of the first $i-1$ requests, where q_k denotes the selected path of request k. Then

$$p_i(q_1, ..., q_{i-1}) = \sum_{j=1}^{L_{i+1}+1} \{x_{i,j} \cdot p_{i+1}(q_1, ..., q_{i-1}, P_{i,j})\}$$

where $L_{i+1} + 1$ denotes the number of available paths of request $i+1$. It is easy to derive the following inequality:

$$p_i(q_1, ..., q_{i-1}) \geq \min_j\{p_{i+1}(q_1, ..., q_{i-1}, P_{i,j})\} \quad (11)$$

We define $p(leaf)$ as the probability that we reach a bad leaf at level K, and $p(leaf) = 0$ when we reach a good leaf and 1 otherwise.

As there exists at least one good solution for BL-SPM by Theorem 6, there exists at least one leaf making the following inequality hold:

$$1 > p_1 > ... > p_K(q_1, ..., q_{K-1}) > p(leaf) = 0$$

As computing all of the conditional probabilities is time-consuming, we relax $p_i(q_1, ..., q_{i-1})$ to $u_i(q_1, ..., q_{i-1})$, which is defined to be an upper bound on the former, i.e.,

$$u_i(q_1, ..., q_{i-1}) \geq \min_j\{u_{i+1}(q_1, ..., q_{i-1}, P_{i,j})\} \quad (12)$$

Next we walk down $\mathcal{T}$ from the root node to a good leaf by continuously decreasing $u_i(\cdot)$. Inspired by the pessimistic estimators in [23], we construct a function u_{root} as follows:

$$u_{root} = e^{t_0\mathcal{I}_S}\prod_{i=1}^{K}\left\{\sum_{j=1}^{L_i}\mu\hat{x}_{i,j}e^{-t_0 v_i} + 1 - \sum_{j=1}^{L_i}\mu\hat{x}_{i,j}\right\} + \sum_{k=1}^{T\cdot N}\left\{e^{-t_k c}\prod_{i=1}^{K}[\sum_{j=1}^{L_i}\mu\hat{x}_{i,j}e^{t_k r_{i,t} I_{i,j,e}} + 1 - \sum_{j=1}^{L_i}\mu\hat{x}_{i,j}]\right\}$$

In the case that $t_0 = \ln\ [1+D(\mathcal{I}_S, \frac{1}{N+1})]$, $t_k = \ln\ [1+\frac{1-\mu}{\mu}]$, and $r_i, v_i \in [0, 1]$ for request i, it is proved that $u_{root} < 1$ holds. With the parameters set as above, we select a path for request i to minimize the value of $u_i(\cdot)$ from level 1 to level K. Suppose we are at level i and we have selected $P_{i,j}$ for request i, then we replace $\hat{x}_{i,j}$ and $\hat{x}_{i,j^*}(j^* \neq j)$ in the expression of u_{root} with 1 and 0, and calculate $u_i(\cdot)$. At each level, we compare values of $u_i(\cdot)$ when using different selected paths to determine the one that minimizes $u_i(\cdot)$.

Based on the characteristics of u_{root} and the decision tree, we propose an algorithm to reach a good leaf thus completing the path selection for each request from the root node of the decision tree. As mentioned above, for a request i, there are $L_i + 1$ available paths to choose from. By selecting different paths for request i, we calculate the values of $u_i(\cdot)$ and finally choose the path that minimizes it. By this method, we continuously decrease the probability of reaching a bad leaf thus we could reach a good leaf at the last level. Since there are K requests, we repeat the above procedures for K times to reach a good leaf. We present our algorithm, tree-based approximation algorithm (TAA), in **Algorithm 2**. Despite there are $\prod_{i=1}^{K}(L_i + 1)$ leaves in the decision tree, from the root node to a good leaf, our algorithm needs to do at most $\sum_{i=1}^{K}(L_i + 1)$ examination. Compared with the brute force method, the computing time is greatly reduced. So far, we have two algorithms, MAA and TAA, to solve RL-SPM and

Algorithm 2: TAA: Tree-based Approximation Algorithm

Input: Requests: $\{1, 2, ..., K\}$; Paths: $\{P_1, P_2, ..., P_K\}$; Bandwidth: $\{c_e\}$;
Output: Transmission paths for requests: $\{x_{i,j}\}$;

```
Normalize the rates and values of requests
Relax BL-SPM and solve its relaxed linear program
Select the scaling factor μ according to (6)
for 1 ≤ i ≤ K do
    for 1 ≤ j ≤ L_i + 1 do
        if fixing x_{i,j} to 1 minimizes u_root then
            x_{i,j} ← 1
        else
            x_{i,j} ← 0
        end
    end
end
return {x_{i,j}}
```

Figure 2. Network topology of B4 [3]

BL-SPM respectively. We deploy them in Metis to find a good schedule for cloud providers to make more profit.

V. Evaluation

In this section, we conduct extensive evaluations to compare Metis and other solutions. We present the evaluation methodology and results in the following.

A. Evaluation methodology

To comprehensively show the performance of our solution and existing schemes, we conduct evaluations on two networks:

- B4: Google's Inter-DC WAN [3], which consists of 12 data centers and 19 bi-directional links (as shown in Fig. 2). It is a typical Inter-DC WAN.
- SUB-B4: a sub-network of B4, which consists of 6 data centers (DC1 $\sim$ DC6) and 7 links between them. It is a small-scale Inter-DC WAN.

We set the bandwidth prices of Inter-DC links based on the relative bandwidth prices provided by Cloudflare [9]. Using 10Gbps as a unit of bandwidth, the bandwidth cost is the product of bandwidth price and bandwidth usage. Following the previous work [6] [20] and the price lists provided by popular cloud providers to cloud tenants, we generate user requests with a synthetic model for a billing cycle consisting of 12 time slots, which represent 12 months. The arrivals of user requests follow Poisson distribution and the bandwidth requirements follow uniform distribution between 0.1Gbps and 5Gbps. The start time and the end time of user requests are randomly set within 12 time slots. For any request, its source data center and destination data center are selected randomly in a specific network (B4 or SUB-B4). We set the value of user request based on the bandwidth requirements and the bandwidth prices published by cloud providers [24]–[26].

We select a set of representative solutions to compare with our solutions: 1) *MinCost*. Using fixed rules in scheduling, it always selects the path with the least bandwidth price (i.e., min-cost path) to deliver traffic data between data centers. In our evaluation, it reserves exclusive bandwidth for users on the min-cost paths. 2) *Amoeba* [20]. It is an Inter-DC flow scheduler to satisfy as many user requests as possible under a fixed amount of bandwidth. 3) *EcoFlow* [17]. It is an economical and deadline-driven flow scheduler. It adopts multipath method to transfer Inter-DC flows with less bandwidth. In our evaluation, it handles user requests one by one and accepts the user requests that generate higher service profits. As the service values of requests are fixed in our problem, it is difficult to compare Metis with the dynamic-pricing based solutions such as Pretium [8]. Thus we have not compared Metis with them. We implement a simulator that integrates the above solutions with C++ and call the Gurobi Optimizer 7.5.2 to solve the LP and ILP problems.

B. Evaluation results

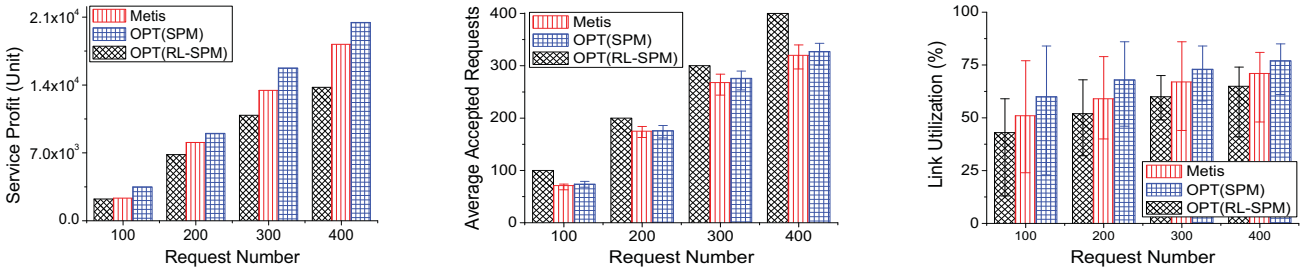
We compare our solution with other solutions under different settings, and show their performance in the following:

1) Solution performance: To evaluate the difference between Metis and the optimal solution of SPM on maximizing service profit and verify the benefit of declining some user requests for higher cloud service profit, we compare Metis with the optimal solution of SPM, denoted as "OPT(SPM)", and the optimal solution with all user requests accepted, denoted as "OPT(RL-SPM)", on the SUB-B4 network. In our evaluation, OPT(RL-SPM) takes the optimal solution to schedule them with the minimum bandwidth cost. We show the comparing results in Fig. 3.

In Fig. 3a, OPT(SPM) makes more service profit than Metis and OPT(RL-SPM) by directly solving the ILP problem to maximize the profit. When there are 400 user requests, it takes more than 1000 seconds to get the optimal request schedule while Metis uses only several hundreds of milliseconds to calculate a feasible schedule. Although the profit of Metis is 11% lower than that of the optimal one using OPT(SPM), it is still 32.3% higher than that of OPT(RL-SPM), which shows the advantage of Metis on making more service profit and the shortage of current service mode on the cloud computing market by accepting all user requests without considering the bandwidth cost.

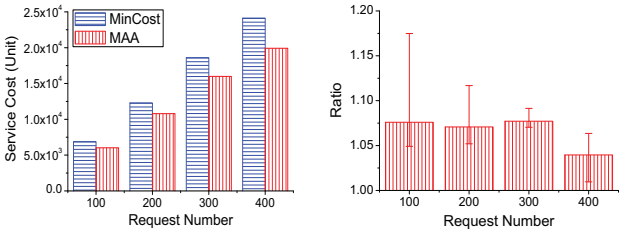
As shown in Fig. 3b, OPT(RL-SPM) accepts all the user requests, while OPT(SPM) and Metis accept only parts of the user requests. To satisfy all requests, OPT(RL-SPM) has to purchase more bandwidth than others, while some of the bandwidth is underused. Without accepting all requests, OPT(SPM) and Metis have larger spaces to search for a better schedule to fully use the purchased bandwidth and make more service profit than OPT(RL-SPM).

In Fig. 3c, we present the link utilization of different solutions with different requests, including the maximum, minimum and average link utilization. OPT(SPM) has the highest average link utilization, while OPT(RL-SPM) has the lowest one by satisfying all user requests even with profit loss. Taking into account the computing time and profit difference between Metis and OPT(SPM), it is more proper to use Metis in practice.

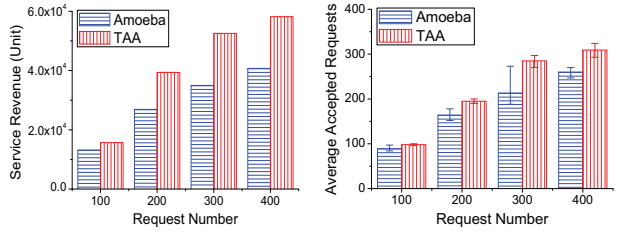


(a) Service profit (b) Accepted request number (c) Link utilization

Figure 3. Metis vs. Optimal solution on SUB-B4



(a) Service cost (b) Ratio



(c) Service revenue (d) Accepted request number

Figure 4. Performance of MAA and TAA on B4

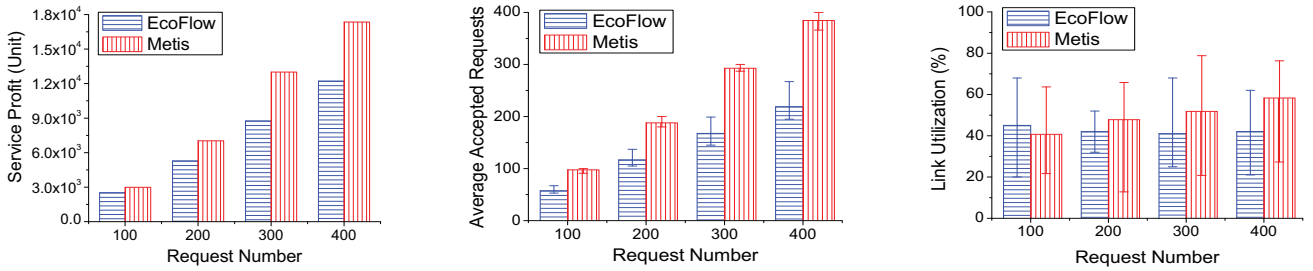
2) Performance of MAA and TAA: Before showing the performance of Metis on maximizing the service profit in geo-distributed clouds, we fist present the performance advantage of MAA and TAA as compared to existing solutions, MinCost and Amoeba [20].

In Fig. 4a, we compare the service cost of MAA and MinCost with different requests on B4 network. To satisfy the same set of requests, the service cost of MinCost is up to 21.1% higher than that of MAA. Scheduling requests based on the fixed policy to reserve bandwidth on the min-cost path without considering the relationship between requests, MinCost suffers from higher bandwidth usage and higher bandwidth cost. Different from MinCost, MAA is an LP-based algorithm. Benefitting from the relax-and-round method and selecting routing path based on the optimal solution of the relaxed RL-SPM problem, it efficiently schedules user requests to make full use of the purchased bandwidth. In Fig. 4a, the difference of service cost between Metis and MinCost increases with the request number. This is because the shortages of the fixed scheduling policy used by MinCost worsens its path selection and increases its bandwidth cost, while MAA has more potential to fully use the purchased bandwidth and generate less bandwidth cost.

In MAA, we adopt randomized rounding method to decide routing paths for user requests based on the optimal solution of relaxed RL-SPM problem. In Fig. 4b, we present the ratio of bandwidth cost with randomized rounding to that of the optimal scheduling in different network settings. For each set of user requests, we repeat the randomized rounding procedure for 1000 times and calculate the bandwidth cost. It shows that, the ratio is always less than 1.2, which demonstrates that our algorithm can achieve a good performance that is comparable to the optimal solution.

As TAA and Ameoba [20] work with fixed bandwidth to satisfy more requests for the maximum service revenue, following the setup in [20], where each link has a uniform capacity, we set the bandwidth of links in the B4 network to 100Gbps, i.e., 10 units of bandwidth. We evaluate the service revenue and the number of satisfied requests under different situations. In Fig. 4c, with the increase of request number, the growth of service revenue of TAA is faster than that of Amoeba. TAA can get up to 50.4% more service revenue than Amoeba. Benefitting from the optimal solution of relaxed BL-SPM and the efficient search method in the decision tree, TAA accepts more user requests than Amoeba with fixed bandwidth. In Fig. 4d, the average number of accepted requests of TAA is up to 33% higher than that of Amoeba. As Amoeba handles requests one by one to accept the ones that can be accommodated by the residual bandwidth without considering future requests, the performance is compromised. Taking into account all user requests, TAA makes more service revenue with a relatively good scheduling solution.

3) Performance of Metis: In Fig. 5, we compare Metis with EcoFlow [17] on the B4 network. In Fig. 5a, the service profit of Metis is up to 32.6% higher than that of EcoFlow. Ecoflow adopts a greedy method to make acceptance decision and it declines too many user requests, while Metis alternately runs MAA and TAA to look for better acceptance and scheduling solutions to maximize the service profit. In Fig. 5b, we compare the numbers of accepted requests in different solutions. The accepted request number of EcoFlow is up to 43.1% less than that of Metis, which leads to less service



(a) Service profit (b) Accepted request number (c) Link utilization

Figure 5. Performance of Metis on B4

revenue and service profit. As shown in Fig. 5c, the average link utilization of Metis is up to 38% higher than that of EcoFlow. This is because Metis accepts more requests to fully uses the purchased bandwidth and make more service profit. From the result, we find that, it is necessary to make the good acceptance and scheduling decisions simultaneously for maximizing the service profit.

VI. Related Work

Traffic engineering for Inter-DC WANs is an attractive topic recently. Benefitting from the emerging software-defined networking, SWAN [2] and B4 [3] are proposed to boost the utilization of Inter-DC WANs by scheduling traffic with centralized controllers. Tempus [27] is proposed to maximize the fraction of transfers delivered before their deadlines, which achieves fairness among all transfers. All of them are cost-unaware and deadline-agnostic solutions. Despite Amoeba [20] has addressed the deadline-agnostic problem and provides deadline guarantees for the accepted transfers, it does not consider the bandwidth cost. In this paper, to maximize the service profit for cloud providers, we alternately minimize the bandwidth cost and maximize the service revenue for many rounds.

Economically scheduling transfers in Inter-DC WANs has also attracted much research attention to reduce the bandwidth cost. NetStitcher [13] and Postcard [14] adopt store-and-forward strategy to deliver data between data centers, where large data transfers are delayed and sent at non-peak hours to fully utilize the already-paid bandwidth and reduce the charge. However, they do not consider deadlines of large transfers, and require large storage in data centers to store data temporarily. EcoFlow [17] splits inter-datacenter flows into multiple paths to avoid the increases of charging volumes and bandwidth cost, which may introduce packet-level reordering problems and degrade the service performance. In contrast, our solution does not need any extra storage in data centers or packet-reordering. We alternately maximize the service revenue by accepting a subset of requests with high service values and minimize the service cost by delivering them over selective paths with low bandwidth prices.

Dynamic pricing is a common method to improve the cost efficiency of Inter-DC WANs. For example, Pretium [8] combines it with traffic engineering to maximize the social welfare, i.e., the total value generated to society (over all served requests) minus the operational cost. It requires cloud providers to modify the current pricing mechanism. Besides, in peak hours, it requires customers to make a concession to either lower their service performance or raise their delivery bids. Requiring neither modification to current pricing mechanism nor complicated negotiation process with customers, our solution can be implemented in current systems to greatly increase the service profits for cloud providers.

VII. Conclusion

The popularity of cloud computing and the emergence of new clouds increase the competition between cloud providers. Maximizing service profit is critical for them to win the business, while it is challenging because of its NP-hardness. In this paper, we first formulate the problem of service profit maximization and prove its NP-hardness. We then propose a framework, Metis, to alternately maximize the service revenue and minimize the service cost to maximize the service profit for cloud providers. We design two approximation algorithms based on randomized rounding techniques and Chernoff-Hoeffding bound and prove their approximation ratios. Our solution outputs the schedule for user requests in polynomial time without incurring too high loss. Extensive evaluations demonstrate that, compared with the existing solutions, Metis increases the service profit for more than 30%.

Acknowledgment

This work is supported by National Key R&D Program of China under Grant 2017YFB1010002 and NSFC (No. 61872211). Zhenjie Yang's research is supported by CSC (No. 201806210244). Xin Wang's research is supported by NSF CNS 1526843. Minming Li's research is supported by a grant from Research Grants Council of the Hong Kong Special Administrative Region, China (No. CityU 11268616) and NSFC (No. 11771365). We would like to thank the anonymous reviewers for their valuable feedback and suggestions.

REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "A view of cloud computing," *Communications of the ACM (2010)*.

[2] C.-Y. Hong, S. Kandula, R. Mahajan, M. Zhang, V. Gill, M. Nanduri, and R. Wattenhofer, "Achieving high utilization with software-driven wan," in *ACM SIGCOMM (2013)*.

[3] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu *et al.*, "B4: Experience with a globally-deployed software defined wan," in *ACM SIGCOMM (2013)*.

[4] Y. Chen, S. Jain, V. K. Adhikari, Z.-L. Zhang, and K. Xu, "A first look at inter-data center traffic characteristics via yahoo! datasets," in *IEEE INFOCOM (2011)*.

[5] Z.-H. Zhan, X.-F. Liu, Y.-J. Gong, J. Zhang, H. S.-H. Chung, and Y. Li, "Cloud computing resource scheduling and a survey of its evolutionary approaches," *ACM CSUR (2015)*.

[6] X. Jin, Y. Li, D. Wei, S. Li, J. Gao, L. Xu, G. Li, W. Xu, and J. Rexford, "Optimizing bulk transfers with software-defined optical wan," in *ACM SIGCOMM (2016)*.

[7] I. Takanori, "Large-capacity optical transmission technologies supporting the optical submarine cable system," *NEC TECHNICAL JOURNAL (2010)*.

[8] V. Jalaparti, I. Bliznets, S. Kandula, B. Lucier, and I. Menache, "Dynamic pricing and traffic engineering for timely inter-datacenter transfers," in *ACM SIGCOMM (2016)*.

[9] R. Nitin, "Bandwidth costs around the world," https://blog.cloudflare.com/bandwidth-costs-around-the-world/.

[10] K. Hsieh, A. Harlap, N. Vijaykumar, D. Konomis, G. R. Ganger, P. B. Gibbons, and O. Mutlu, "Gaia: Geo-distributed machine learning approaching lan speeds." in *USENIX NSDI (2017)*.

[11] 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 NSDI (2015)*.

[12] "First-price sealed-bid auction," https://en.wikipedia.org/wiki/First-price_sealed-bid_auction.

[13] N. Laoutaris, M. Sirivianos, X. Yang, and P. Rodriguez, "Inter-datacenter bulk transfers with netstitcher," *ACM SIGCOMM (2011)*.

[14] Y. Feng, B. Li, and B. Li, "Postcard: Minimizing costs on inter-datacenter traffic with store-and-forward," in *IEEE ICDCSW (2012)*.

[15] Y. Wang, S. Su, A. X. Liu, and Z. Zhang, "Multiple bulk data transfers scheduling among datacenters," *Elsevier Computer Networks (2014)*.

[16] W. Li, K. Li, D. Guo, G. Min, H. Qi, and J. Zhang, "Cost-minimizing bandwidth guarantee for inter-datacenter traffic," *IEEE Transactions on Cloud Computing (2016)*.

[17] Y. Lin, H. Shen, and L. Chen, "Ecoflow: An economical and deadline-driven inter-datacenter video flow scheduling system," in *ACM International Conference on Multimedia (2015)*.

[18] Z. Zhang, M. Zhang, A. G. Greenberg, Y. C. Hu, R. Mahajan, and B. Christian, "Optimizing cost and performance in online service provider networks." in *USENIX NSDI (2010)*.

[19] M. R. Gary and D. S. Johnson, "Computers and intractability: A guide to the theory of np-completeness," (1979).

[20] H. Zhang, K. Chen, W. Bai, D. Han, C. Tian, H. Wang, H. Guan, and M. Zhang, "Guaranteeing deadlines for inter-datacenter transfers," in *ACM EuroSys (2015)*.

[21] P. Raghavan and C. D. Tompson, "Randomized rounding: a technique for provably good algorithms and algorithmic proofs," *Springer Combinatorica (1987)*.

[22] V. Guruswami, S. Khanna, R. Rajaraman, B. Shepherd, and M. Yannakakis, "Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems," *Elsevier Journal of Computer and System Sciences (2003)*.

[23] P. Raghavan, "Probabilistic construction of deterministic algorithms: approximating packing integer programs," *Elsevier Journal of Computer and System Sciences (1988)*.

[24] "Aliyun," https://cn.aliyun.com.

[25] "Azure," https://azure.microsoft.com/.

[26] "Aws," https://aws.amazon.com/.

[27] S. Kandula, I. Menache, R. Schwartz, and S. R. Babbula, "Calendaring for wide area networks," in *ACM SIGCOMM (2014)*.