# Cost-Efficient Scheduling of Bulk Transfers in Inter-Datacenter WANs

Zhenjie Yang, Yong Cui, Xin Wang, Yadong Liu, Minming Li, Shihan Xiao, and Chuming Li

***Abstract*— With the quick growth of traffic between data centers, inefficient transfer scheduling in inter-datacenter networks can lead to a huge waste of bandwidth thus significant bandwidth cost. Previous work have explored different ways, such as software-defined WANs and dynamic pricing mechanisms, to overcome the inefficiency of inter-datacenter networks. However, there is a big challenge in addressing the fundamental conflicts between the deadline-aware transfer scheduling and minimizing the bandwidth cost. Unlike existing efforts that schedule inter-datacenter transfers under fixed link capacities, wherein some deadlines are violated and the service quality is degraded, we aim to finish *all* the transfers on time with as little bandwidth as possible to minimize the bandwidth cost. We take into account the variation of bandwidth price and the deadline requirements of services, and formulate the problem of cost-efficient scheduling of bulk transfers with deadline guarantee, which is shown to be NP-hard. Benefitting from the relax-and-round method, we propose a progressively-descending algorithm (PDA) to schedule bulk transfers and meet the above goals with a guaranteed approximation ratio. We apply our algorithm in a bulk transfer scheduler, Butler, and build a small-scale testbed to evaluate its efficiency. Both large-scale simulation and testbed experiment results validate the ability of our scheme on cutting down the bandwidth cost. Compared with existing approaches, it reduces up to 60% bandwidth cost and increases the network utilization by up to 140%.**

***Index Terms*— Inter-datacenter WAN, cost, scheduling.**

## I. Introduction

MANY online service providers and cloud platform providers own data centers across different geographic zones and run globally-distributed applications in their data centers [1]–[7]. Inter-datacenter wide area network (Inter-DC WAN) is the critical infrastructure to connect these geographically distributed data centers. Except for the providers that build dedicated lines between their data centers [2], [3], [8], service providers generally need to lease bandwidth from Internet service providers (ISPs) to deliver the traffic among data centers over Inter-DC WAN. Typically, the bandwidth is sold at a fixed price per unit (e.g., 10Gbps and 40Gbps) [2], [9], [10] with the price varying across different regions of the world [11]. The bandwidth usage is calculated over a fixed *billing cycle*, such as a day, a week or a month [9], [12]. Generally, data center operators pay hundreds of millions of dollars per year to ISPs for traffic delivery over their WANs [2], [3], [8]. As reported in [13], the network cost accounts for 15% of the total expenditure in data centers, which is comparable to the power cost in data centers [14].

Data center operators, however, are not able to gain the full return from their investments. Even for the very busy links, the average link utilization is only 40-60% [2]. In most cases, there is no coordination among transmissions from different services over the same Inter-DC WAN, and data is sent across the network whenever available and as much as the services need. This kind of "casual" scheduling of Inter-DC transfers gives rise to a tremendous wastage of bandwidth resources and incurs a high bandwidth cost for data center operators [2], [3].

Traffics over Inter-DC WAN are generated by various globally-distributed applications. Among all the data delivered over Inter-DC WAN, bulk transfers have large sizes (e.g., several TBs to PBs) and account for a large proportion (e.g., 85-95% [4], [15]) of the total traffic. There are various types of bulk transfers in Inter-DC WANs. As some typical examples, search engines synchronize search indices across all data centers periodically [3], [16], [17]; Financial institutions back up their daily transaction records in remote sites every trading day [15]; Video content publishers release the latest videos across geographical regions to viewers [18]–[20]. Bulk transfers have relative long deadlines, e.g., several hours to days [15]. Transmitting the data timely and meeting the expected deadlines of services are essential for service providers to win the business. As a large fraction of the traffic in Inter-DC WANs is bulk transfer with long deadline, there is a great potential for data center operators to cut down the bandwidth cost with the efficient scheduling of bulk transfers.

Data center operators face two challenges in scheduling bulk transfers over Inter-DC WANs: (1) Transferring bulk transfers in non-peak hours vs. Guaranteing their deadlines. As bulk

Manuscript received December 24, 2018; accepted July 30, 2019; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor S. Uhlig. Date of publication August 26, 2019; date of current version October 15, 2019. This work was supported in part by the National Key R&D Program of China under Grant 2018YFB1800303 and in part by the NSFC Project under Grant 61872211. The work of Z. Yang was supported by the China Scholarship Council under Grant 201806210244. The work of X. Wang was supported by the NSF CNS under Grant 1526843. The work of M. Li was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region, China, under Grant CityU 11268616 and in part by the NSFC under Grant 11771365. *(Corresponding author: Yong Cui.)*

Z. Yang, Y. Cui, Y. Liu, and C. Li are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: yangzj15@mails.tsinghua.edu.cn; cuiyong@tsinghua.edu.cn; lichuming.lcm@gmail.com; liuyd17@mails.tsinghua.edu.cn).

X. Wang is with the Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY 11790 USA (e-mail: x.wang@stonybrook.edu).

M. Li is with the Department of Computer Science, City University of Hong Kong, Hong Kong, and also with the Shenzhen Research Institute, City University of Hong Kong, Hong Kong (e-mail: minming.li@cityu.edu.hk).

S. Xiao is with Huawei Technologies Co., Ltd., Beijing 100080, China (e-mail: xiaoshihan@huawei.com).

Digital Object Identifier 10.1109/TNET.2019.2934896

transfers have long deadlines, data center operators could wait and send them in non-peak hours to lower the bandwidth usage, but at the risk of violating their deadlines. (2) Making full use of complete units of bandwidth. As ISPs sell Inter-DC bandwidth in high prices per unit, which is typically 10Gbps and 40Gbps [10], scheduling bulk transfers to fully use the purchased units of bandwidth is essential for cutting down the total bandwidth cost. Bandwidth prices are largely different across the world, e.g., the highest bandwidth price is up to 21.3 times higher than the lowest one [21], further increasing the difficulty of transferring bulk data with less bandwidth cost. For data center operators, it is hard to determine *at what rate and on which path to send data* could meet the requirements of bandwidth cost and the deadlines of bulk transfers simultaneously.

Recently, various efforts have been made to improve the network utilization or cost efficiency of Inter-DC WANs. Existing solutions often attempt to improve the network utilization without considering the transmission cost and deadline [2], [3], [16] or only consider the deadline [4]. As there is a large proportion of bulk data in Inter-DC WANs and the cost of Inter-DC transfers is high, some solutions equip data centers with large storages and adopt the store-and-forward approach to schedule bulk transfers [8], [22], which lead to high storage costs for data center operators. Overall, service providers are caught in a dilemma when to apply which of these solutions in real systems.

In this work, we take the initiative to efficiently schedule bulk transfers with low bandwidth cost while guaranteeing their deadlines. To achieve the goal, we formulate the problem of cost-efficient scheduling of bulk transfers (CESBT), taking into account the difference in bandwidth prices and the integer property of bandwidth charging. We show our problem is NP-hard, and propose a progressively-descending algorithm (PDA) that is derived from the relax-and-round technique to schedule bulk transfers in cost-efficient manner while guaranteeing the deadlines with a guaranteed approximation ratio. We apply PDA to the popular architecture of software-defined Inter-DC WAN, and design a bulk transfer scheduler, Butler, for Inter-DC WANs. To evaluate the effectiveness of our scheme, besides simulations, we implement Butler and test its performance over a small-scale testbed. Following [4], [9], [15], we generate workload with a synthetic model and the arrival rate of bulk transfer requests follows Poisson distribution. Our extensive simulations and testbed results demonstrate that our scheme can reduce more than half the bandwidth cost annually for data center operators.

The remainder of this paper is organized as follows: The background and motivation are presented in Section II. The formulation of CESBT and PDA is introduced in Section III. The Butler is introduced in Section IV. The evaluation results are presented in Section V, followed by the related work in Section VI and the conclusion in Section VII.

## II. Background and Motivation

In this section, we introduce the basic architecture of software-defined Inter-DC WAN and give a motivating example to show the advantages of optimizing the scheduling of Inter-DC transfers.

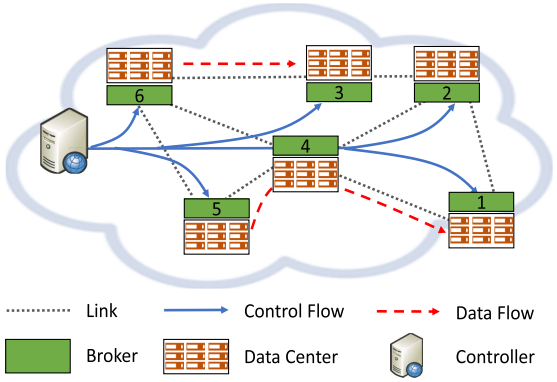



Fig. 1. Software-defined Inter-DC WAN.

### A. Software-Defined Inter-DC WAN

Nowadays, emerging applications and operational scenarios raise exacting requirements on data transmission inside clouds [23]. As SDN shows great promise in constructing high-performance networks, to satisfy the strict requirements posed by cloud services, many data center operators exploit SDN technologies to efficiently schedule transfers over Inter-DC WANs, also known as *software-defined Inter-DC WAN* [2]–[4], [23]. Fig. 1 shows a typical architecture of software-defined Inter-DC WANs. Besides data centers and links, to facilitate the use of SDN, it includes a controller and multiple brokers each associated with a data center. Typically, the controller is responsible for scheduling transfers and handling errors. Brokers send the flow information and network status to the controller to determine the schedule of transfers, and enforce rate allocation inside data centers following the instruction of the controller. In software-defined Inter-DC WANs, control flows are applied to exchange messages between the controller and brokers, and data flows carry the data generated by services.

### B. Motivating Example

As a large fraction of the traffic in Inter-DC WANs is bulk transfer with long deadline, optimizing bulk transfers has a great potential for data center operators to cut down the bandwidth cost. We give an example to show the advantage. In Fig. 2a, the Inter-DC WAN connects three data centers (DC1, DC2 and DC3) with three bidirectional links (link1, link2 and link3). In Fig. 2b, three bulk transfers R1, R2 and R3 are associated with different arrival time ($arr.$), deadlines ($ddl.$) and demands ($dem.$), with $X \rightarrow Y$ indicating the transfer of data from DC-$X$ to DC-$Y$. Generally, the bandwidth price (i.e., the cost per unit of bandwidth) varies with links [11] and the bandwidth cost is equal to the product of the bandwidth consumed in one billing cycle and the bandwidth price [24]. In our example, the billing cycle is set to be between the slots 1 and 10. Fig. 2c gives bandwidth prices on different links.

We adopt three scheduling solutions to finish these transfers on time and compare their total bandwidth costs. In Fig. 2d,

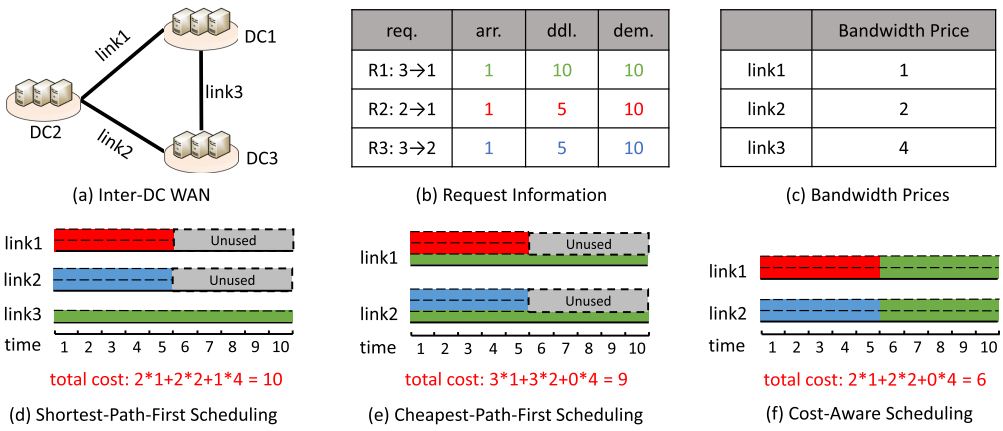



Fig. 2. Example: comparing different scheduling solutions.

Shortest-Path-First (SPF) delivers each transfer on the shortest path. It delivers R1, R2 and R3 on the paths "DC3-DC1", "DC2-DC1" and "DC3-DC2", with the transmission rate equal to the transfer demand divided by the maximum tolerable transfer time [22]. According to Fig. 2b, the desired transmission rates of R1, R2 and R3 are 1, 2 and 2 respectively. Thus the bandwidth consumed on link1, link2 and link3 is 2, 2 and 1. Multiplied by the bandwidth prices, the total cost is $2 * 1 + 2 * 2 + 1 * 4 = 10$. In Fig. 2e, Cheapest-Path-First (CPF) delivers each transfer on the path with the lowest bandwidth price. It delivers R1, R2 and R3 on the paths "DC3-DC2-DC1", "DC2-DC1" and "DC3-DC2" respectively. As the desired transmission rates of R1, R2 and R3 are 1, 2 and 2, the bandwidth consumed on link1 and link2 is 3 and 3, and the total cost is $3 * 1 + 3 * 2 = 9$, which is slightly lower than that of SPF.

Different from the above solutions, Cost-Aware (CA) schedules these transfers with the least bandwidth cost. As R1 has a longer deadline than R2 and R3 and it can use the links that deliver R2 and R3, CA selects the paths "DC3-DC2-DC1", "DC2-DC1" and "DC3-DC2" for R1, R2 and R3 respectively. It starts to deliver R1 after R2 and R3 have already finished. By this way, CA sends R2 and R3 at the rate of 2 between time slots 1 and 5 and sends R1 at the rate of 2 between time slots 6 and 10. In this case, the bandwidth consumed on link1 and link2 is 2 and 2, and the total cost is only $2*1+2*2 = 6$. Compared with SPF and CPF, CA reduces the bandwidth cost by 40% and 33% respectively. This demonstrates the benefit of optimally scheduling the transfers over Inter-DC WANs to reduce the bandwidth cost.

## III. Cost-Efficient Scheduling of Bulk Transfers

In this section, we first formulate the problem of CESBT and prove its NP-hardness. Then we present the design details of our algorithm and prove its approximation ratio.

### A. Problem Formulation

Before our formulation, we introduce the necessary definitions and notations related with our goal in the following:

*1) Inter-DC WAN:* The network topology is represented as a directed graph $G = (V, E)$, where $V$ denotes the set of data centers and $E$ denotes the set of links. Each link $e \in E$ denotes a directed link between a data center pair.

*2) Bandwidth Cost:* We consider a billing cycle consisting of $|T|$ independent time slots, denoted as $T = \{1, \ldots, |T|\}$. According to [22], [24], the bandwidth cost of link $e$ is a non-decreasing function of the bandwidth charge $c_e$, which is an integer in practice [9]. If per unit bandwidth on link $e$ is charged at $u_e$, the cost of $e$ can be calculated as the product of $u_e$ and $c_e$ [22], [24].

*3) Bulk Transfer:* In a billing cycle, there are multiple bulk transfer requests, each of which is specified by $f_i = \{s_i, t_i, d_i, a_i, \tau_i\}$, where $s_i,\ t_i \in V$, $s_i \neq t_i$, $a_i,\ \tau_i \in T$, $a_i \leq \tau_i$. A bulk transfer $f_i$ arriving at time slot $a_i$ with the traffic demand $d_i$ is requested to be sent from the source data center $s_i$ to the destination data center $t_i$ before its deadline $\tau_i$. The maximum tolerable transfer time of $f_i$ is defined as the duration between its start time $a_i$ and its deadline $\tau_i$. We use $F = \{f_1, \ldots, f_n\}$ to denote the set of bulk transfers, and $I = \{1, \ldots, n\}$ to denote the index set of $F$. We use $f_i$ and $i$ to denote the $i$-th bulk transfer interchangeably.

In our formulation of **C**ost-**E**fficient **S**cheduling of **B**ulk **T**ransfers (CESBT) over Inter-DC WAN, the objective is formally expressed as:

$$\min \sum_{e \in E} u_e \times c_e$$

which is subject to the following constraints:

*4) Flow Conservation Constraints:* For any bulk transfer $i \in I$ with the flow $f_i$, if a data center is neither its source $s_i$ nor its destination $t_i$, the volume of outgoing traffic of $f_i$ should be equal to its incoming traffic. Let $x_e^i(t)$ denote the volume of the part of traffic of $f_i$ that transmits through the link $e$ in the time slot $t$, then we have

$\forall i \in I,\ v \in V \backslash \{s_i, t_i\},\ t \in [a_i, \tau_i]$ :

$$\sum_{e \in \delta^-(v)} x_e^i(t) - \sum_{e \in \delta^+(v)} x_e^i(t) = 0 \qquad (1)$$

TABLE I
NOTATIONS AND DEFINITIONS

| Notations | Definitions |
|---|---|
| $G$ | $(V, E)$: the topology of Inter-DC WAN |
| $F$ | the set of bulk transfers |
| $I$ | the set of indices of bulk transfers |
| $J$ | the value of rounding-depth |
| $K$ | the value of rounding-span |
| $M$ | the total bandwidth cost |
| $T$ | the set of indices of the time slots |
| $U$ | the unit of charging bandwidth |
| $u_e$ | the cost of per unit of charging bandwidth on $e$ |
| $c_e$ | the charging bandwidth of $e$ |
| $f_i$ | $\{s_i, t_i, a_i, \tau_i, d_i\}$: the $i$-th bulk transfer. $s_i$ : source; $t_i$: destination; $a_i$: arrival time; $\tau_i$: deadline; $d_i$: demand. |
| $x_e^i(t)$ | the traffic volume of $f_i$ that flows through $e$ at $t$ |
| $\delta^-(v)$ | the set of incoming links of data center $v$ |
| $\delta^+(v)$ | the set of outgoing links of data center $v$ |

where $\delta^-(v)$ and $\delta^+(v)$ denote the set of incoming links and outgoing links of data center $v$, respectively. For clarity, we summarize the notations to use in this section in Table I.

*5) Transmission Time Constraints:* For any bulk transfer $f_i$ ($i \in I$) with the demand $d_i$, in order to complete the transfer without violating its deadline, the volume of the outgoing traffic from its source $s_i$ between its arrival time $a_i$ and its deadline $\tau_i$ should be equal to $d_i$. Similarly, the volume of incoming traffic of $f_i$ to its destination $t_i$ during $a_i$ and $\tau_i$ should be equal to $d_i$. Thus, the following constraints should be satisfied:

$$\sum_{t\in[a_i,\tau_i]}\Big(\sum_{e\in\delta^+(s_i)} x_e^i(t) - \sum_{e\in\delta^-(s_i)} x_e^i(t)\Big) = d_i, \quad \forall i \in I \tag{2}$$

$$\sum_{t\in[a_i,\tau_i]}\Big(\sum_{e\in\delta^-(t_i)} x_e^i(t) - \sum_{e\in\delta^+(t_i)} x_e^i(t)\Big) = d_i, \quad \forall i \in I \tag{3}$$

Considering each transfer could be routed along multiple paths, we assume existing techniques such as [25] and [26] can solve the problem with possible packet-level reordering.

*6) Bandwidth Charge Constraints:* For any link $e \in E$, its bandwidth charge $c_e$ allows for the transferring of $||c_e||$ units in one time slot, which should be equal to or greater than the volume transferred in any time slot:

$$\sum_{i\in I} x_e^i(t) \le ||c_e||, \quad \forall e \in E,\ t \in T \tag{4}$$

For simplicity, we denote the total bandwidth cost as $M$:

$$M \triangleq \sum_{e\in E} u_e \times c_e.$$

Thus our optimization goal is as follows:

$$\min\ M \tag{5}$$

$$\text{s.t.}\ \ (1),\ (2),\ (3),\ (4)$$

$$x_e^i(t) \ge 0, \quad \text{for all } i,\ e,\ t \tag{6}$$

$$c_e \in \mathbb{Z}^+, \quad \text{for all } e \tag{7}$$

Different from the previous work [2]–[4], [16] that attempts to improve the network utilization without considering the transmission cost and deadline or only consider the deadline, we take into account the service performance and operation expenditure simultaneously in our formulation, and aim to efficiently schedule bulk transfers with low bandwidth cost while guaranteeing their deadlines. By reducing the minimum cost capacity installation (MCCI) problem [27] to a special case of CESBT, we obtain the following theorem:

*Theorem 1: CESBT is NP-hard.*

*Proof:* CESBT contains the MCCI problem as a special case, which is known to be strongly NP-hard in general graphs [27]. Consider a directed graph $G = (V, E)$, where $V$ denotes the set of nodes and $E$ denotes the set of links. Given a set of demands $\{d_k | k = 1, \ldots, \mathcal{K}\}$, each of which specified by an ordered pair of nodes $(s_k, t_k)$, $s_k, t_k \in V$ and $s_k \ne t_k$, each pair $(s_k, t_k)$ corresponds to a commodity flow to be sent from the source node $s_k$ to the destination node $t_k$ using links in $E$. The MCCI problem can be defined as:

$$\min\ \sum\nolimits_{e_{ij}\in E} c_{ij}^* \cdot y_{ij}^*$$

$$s.t.\ \begin{cases} \sum_j x_{ji}^k - \sum_j x_{ij}^k = d_k, & i = t_k,\ \forall k \\ \sum_j x_{ji}^k - \sum_j x_{ij}^k = 0, & i \notin \{s_k, t_k\},\ \forall k \\ \sum_{k\in[1,\mathcal{K}]} x_{ij}^k \le y_{ij}^*, & \forall e_{ij} \in E \\ x_{ij}^k \ge 0, y_{ij}^* \in N^+, & \forall e_{ij} \in E \end{cases}$$

where $y_{ij}^*$ denotes the capacity requirement on $e_{ij}$, which is the directed link from $i$ to $j$, $c_{ij}^*$ denotes the cost of obtaining one unit of capacity on $e_{ij}$, and $x_{ij}^k$ denotes the amount of the $k$-th transfer flowing through $e_{ij}$. The objective of the MCCI problem is to obtain a minimum cost installation of capacities to ensure that all commodities can be shipped simultaneously.

To transform the MCCI problem to a special instance of CESBT, we first construct a special input of CESBT. Consider that all bulk transfers arrive at the beginning of the first time slot, and they must be finished in the end of the first time interval, i.e., $\forall f_i \in F, a_i = 1, \tau_i = 1$. For each link $e_{ij} \in E$, we consider setting the unit price of capacity $u_e$ in the original problem as the unit capacity cost $c_{ij}^*$ in the MCCI problem, where $e$ is the link from $i$ to $j$, then $u_e = c_{ij}^*$. After these settings in polynomial time, an instance of CESBT is constructed. If we can solve CESBT with a polynomial time algorithm, we would obtain the required bandwidth for each link, which is equal to the unit number of capacity in the MCCI problem. In this way, the MCCI problem can also be solved. Therefore, CESBT is at least as hard as the MCCI problem, which is known to be NP-hard. This completes the proof. ■

Since CESBT is NP-hard, there exists no efficient way to find the optimal solution in the polynomial time. As it concerns us, one important question is: *can we develop a polynomial-time algorithm to solve CESBT with a guaranteed approximation ratio?* The key challenges come from the integer nature of the bandwidth charge for each link, the continuous nature of the volume transferred, and their coupling. Specifically, the bandwidth charge $c_e$ is related to both the design objective, i.e., minimize bandwidth cost, and the transferred volume variable

$x_e^i(t)$, which makes it difficult to obtain good result with a reasonable amount of computational time and memory.

### *B. Algorithm Design*

To answer the above question, we propose a progressively-descending algorithm (PDA) to schedule the bulk transfers in a billing cycle. It is run for several rounds following a rounding strategy to reduce the bandwidth cost. There are two important parameters, *rounding-depth* (R-Depth) and *rounding-span* (R-Span), in PDA to determine the bandwidths to charge for different links in each time slot. R-Depth limits the number of rounds to run, and R-Span limits the number of fractional numbers (each representing a charging bandwidth) to be rounded each time and serves as a rounding strategy. Both of them are positive and set by data center operators according to actual needs before running PDA. For simplicity, we use $J$ and $K$ to denote R-Depth and R-Span respectively. Our algorithm consists of four basic operations as follows:

*1) Relaxation:* In each time slot, we have a list of bulk transfers to be scheduled. Our goal is to determine the transferring that can be carried out and the bandwidth to charge over each link. Since bandwidth for charging is an integer in practice, we first relax the integer constraints to linear ones, and make (5) solvable in polynomial time. The relaxed problem is called "Relaxed-CESBT".

*2) Initialization:* We solve Relaxed-CESBT, and obtain a list of fractional charging bandwidths. When we round up all of them, we can easily obtain the upper bound of $M$, which we set as the initial value of $M$.

*3) Rounding:* Among all the obtained fractional charging bandwidths in Initialization, we select $K$ charging bandwidths that are closest to their round-up or round-down, and call them "candidates". We round these candidates to their nearest integers, i.e., their round-up or round-down.

*4) Fixing:* We fix the values of candidates and solve Relaxed-CESBT to find the optimal values on the bandwidth to charge for each link and the transfer allocation. We round the fractional bandwidth for charging to their round-up, and calculate the bandwidth cost, denoted as $obj$. If it is lower than $M$, we update $M$ with $obj$ and carry out Rounding and Fixing in the same way. Otherwise, we remove the first candidate and recover it to its original value. Then we select a new candidate for $K$ and solve Relaxed-CESBT once again.

Before stopping the iterations in PDA, we will repeat the above operations for $J$ rounds, unless the bandwidth cost has not been lowered in a complete round. When PDA is running, we use a counter to record the number of rounds. It is set to zero at the beginning of PDA. We show the pseudocode of PDA in **Algorithm 1** and prove its approximation ratio in the following subsection. As the objective of PDA is to use less bandwidth to transfer data across data centers, it can be also used in the clouds where data center operators run their own backbone and have invested the capital for a fixed backbone. By running with our algorithm, they can get more available bandwidth for other transfers and applications.

**Algorithm 1** PDA: Progressively-Descending Algorithm

**Input**: $F$: the set of bulk transfers; $G$: the topology of network; $\{u_e\}$: the set of bandwidth prices;
**Output**: the rate allocations of each transfer $\{x_e^i(t)\}$; the charging bandwidths of links $\{c_e\}$;

```
Relax the integer constraints of (5);
Initialize M and counter;
while counter is less than J do
    Round K candidates;
    Fix the candidates, and calculate obj;
    if obj < M then
        Update M and counter;
    else
        Update candidates;
    end
    if no candidate is available then
        Break;
    end
end
return rate allocations and charging bandwidths
```

TABLE II
NOTATIONS AND DESCRIPTIONS OF PROBLEMS

| Notation | Problem Description | Optimal Solution |
|---|---|---|
| $\mathcal{P}_0$ | The original CESBT problem with integer charging bandwidth | $\Omega_0$ |
| $\mathcal{P}_1$ | The relaxed-CESBT problem with continuous charging bandwidth | $\Omega_2$ |
| $\mathcal{P}_2$ | The relaxed $\mathcal{P}_1$ problem with $a_i = 1$ and $\tau_i = \|T\|$ for any $i \in I$ | $\Omega_3$ |

### *C. Algorithm Analysis*

In this subsection, we will prove the approximation ratio of PDA. For simplicity, we denote the original CESBT problem as $\mathcal{P}_0$ and the relaxed-CESBT problem as $\mathcal{P}_1$. We further relax the relaxed-CESBT problem and obtain a new problem $\mathcal{P}_2$ as follows: for each bulk transfer $i \in I$, its arrival time $a_i$ equals 1, and deadline $\tau_i$ equals $|T|$. We denote the optimal bandwidth cost of $\mathcal{P}_2$ as $\Omega_3$. For clarity, we list all the related problems in Table II.

For each bulk transfer $i \in I$, there may be multiple available routing paths between $s_i$ and $t_i$, and the cost of per unit of bandwidth on the path $p_i$ is represented as

$$U_{p_i}^i \triangleq \sum_{e \in p_i} u_e$$

Let $p_i^*$ denote the path with the minimum cost of unit bandwidth (termed "min-cost routing path"), and $v_i$ denote the average bandwidth that $i$ requires in problem $\mathcal{P}_2$, i.e.,

$$v_i = \frac{d_i}{||T||},$$

where $d_i$ denotes the size of $i$ and $||T||$ represents the duration of a billing cycle. Then we have the lemma below:

*Lemma 1:* $\Omega_3 = \sum_{i\in I}(U^i_{p^*_i} \times v_i)$.

*Proof:* For each bulk transfer $i \in I$, let $\mathcal{S}_i$ denote the set of paths between $s_i$ and $t_i$. For any feasible solution of problem $\mathcal{P}_2$, let $\overline{E}$ denote the links used by this solution, we have the bandwidth cost as:

$$\begin{aligned} M &= \sum_{e\in\overline{E}} u_e \max_t \sum_{i\in I} ||x^i_e(t)|| \\ &\geq \sum_{e\in\overline{E}} u_e \sum_{i\in I}\sum_{t\in T} \frac{||x^i_e(t)||}{|T|} \\ &= \sum_{e\in\overline{E}} u_e \sum_{i\in I} ||\overline{x}^i_e|| \\ &= \sum_{i\in I}\sum_{e\in\overline{E}} u_e ||\overline{x}^i_e|| \end{aligned}$$

where $||x^i_e(t)||$ denotes the sending rate of $i$ at time $t$ on link $e$. Its value is equal to that of $x^i_e(t)$. We use $||\overline{x}^i_e||$ to denote the average sending rate of $i$ on link $e$ in the billing cycle. Similarly, we denote $||\overline{x}^i_p||$ as the average sending rate of $i$ on path $p$ in the billing cycle. Based on the above definitions, we have

$$\begin{aligned} \sum_{i\in I}\sum_{e\in\overline{E}} u_e||\overline{x}^i_e|| &= \sum_{i\in I}\sum_{p\in\mathcal{S}_i} U^i_p||\overline{x}^i_p|| \\ &\geq \sum_{i\in I}\sum_{p\in\mathcal{S}_i} U^i_{p^*_i}||\overline{x}^i_p|| \\ &= \sum_{i\in I} U^i_{p^*_i}\sum_{p\in\mathcal{S}_i} ||\overline{x}^i_p|| \\ &= \sum_{i\in I} U^i_{p^*_i}\frac{d_i}{||T||} \end{aligned}$$

Thus the following inequations:

$$\begin{aligned} M &\geq \sum_{i\in I}\sum_{p\in\mathcal{S}_i} U^i_p \times ||\overline{x}^i_p(t)|| \\ &\geq \sum_{i\in I} U^i_{p^*_i} \times v_i \end{aligned}$$

hold, and the optimal solution $\Omega_3$ of $\mathcal{P}_2$ is $\sum_{i\in I}(U^i_{p^*_i} \times v_i)$. This completes the proof. ∎

Based on Lemma 1, the optimal solution of problem $\mathcal{P}_2$ can be achieved by setting the sending rate of each transfer as $d_i/||T||$ and selecting the min-cost routing path for each transfer.

We define the *interesting* subset of links as $E^*$: for arbitrary link $e \in E$, if $e$ is in the min-cost routing path of an ordered pair of nodes $(s,t)$, $s,t \in V$, then $e \in E^*$. Based on the definition, we have the following lemma:

*Lemma 2: If there exist bulk transfers between any ordered pair of nodes (s,t), $s,t \in V$, and the set of links used in the optimal solution of $\mathcal{P}_2$ are denoted as $E'$, then we have $E' = E^*$.*

*Proof:* For arbitrary link $e \in E'$, there exists at least one bulk transfer $i \in I$ to make $e$ in the min-cost routing path of itself, i.e., $e$ is in the min-cost routing path of $(s_i, t_i)$, $e \in E^*$ holds, thus $E' \subseteq E^*$ holds.

For arbitrary link $e \in E^*$, there exists at least an ordered pair of nodes $(s,t)$ to make $e$ in the min-cost path of $s$ and $t$. Meanwhile, there exists at least one bulk transfer $i$ whose source is $s$ and destination is $t$, i.e., $s_i = s$ and $t_i = t$. Then $e \in E'$ holds, thus $E^* \subseteq E'$.

Since $E' \subseteq E^*$ and $E^* \subseteq E'$ hold simultaneously, $E' = E^*$ holds. This completes the proof. ∎

*Lemma 3: If the set of links used in the optimal solution of $\mathcal{P}_1$ are denoted as $\hat{E}$, then we have $\hat{E} \subseteq E^*$.*

*Proof:* For arbitrary link $e \in \hat{E}$ but $e \notin E^*$, $e$ is not in the min-cost routing path of any ordered pair of nodes $(s,t)$. Without loss of generality, $e$ connects the source node $s_e$ and the destination node $t_e$. There exists a routing path $p$ whose cost is lower than $e$ and connects $s_e$ and $t_e$. Transferring all the traffic flowing through $e$ to $p$ will reduce the total cost, thus obtain a better solution, which conflicts with the statement that $\hat{E}$ is the set of links used in the optimal solution of $\mathcal{P}_1$. Hence there is no link that belongs to $\hat{E}$ but not $E^*$, and $\hat{E} \subseteq E^*$ holds. This completes the proof. ∎

Combining Lemma 1, 2 and 3, we prove the approximation ratio of PDA and show the following theorem:

*Theorem 2: Let $R$ denote the approximation ratio between the bandwidth cost obtained by PDA and the optimal one and $U$ denote the unit of bandwidth, then*

$$R < 1 + \max_{i\in I}\frac{U}{v_i}$$

*holds.*

*Proof:* With the definitions of $\Omega_0$, $\Omega_2$ and $\Omega_3$ in Table II, we denote the bandwidth cost obtained by PDA as $\Omega_1$, and the bandwidth cost derived from rounding up the fractional bandwidth for charging in the optimal solution of $\mathcal{P}_2$ as $\overline{\Omega}_2$. The link set that corresponds to $\Omega_2$ is denoted by $\hat{E}$ and the link set that corresponds to $\Omega_3$ is denoted by $E'$. It is easy to prove that

$$\frac{\Omega_2}{\Omega_0} \leq 1$$

and

$$\frac{\Omega_1}{\Omega_0} \leq \frac{\overline{\Omega}_2}{\Omega_0}$$

hold. Based on the definitions of $\Omega_2$ and $\overline{\Omega}_2$, we have

$$\begin{aligned} \frac{\Omega_1}{\Omega_0} &< \frac{\Omega_2}{\Omega_0} + \frac{\sum_{e\in\hat{E}} u_e \times U}{\Omega_0} \\ &\leq 1 + \frac{\sum_{e\in\hat{E}} u_e \times U}{\Omega_0} \\ &\leq 1 + \frac{\sum_{e\in\hat{E}} u_e \times U}{\Omega_3} \\ &= 1 + \frac{\sum_{e\in\hat{E}} u_e \times U}{\sum_{i\in I}(v_i \times U^i_{p^*_i})} \\ &\leq 1 + \max_{i\in I}\frac{U}{v_i}\frac{\sum_{e\in\hat{E}} u_e}{\sum_{i\in I} U^i_{p^*_i}} \end{aligned}$$

According to Lemma 2 and Lemma 3, we have

$$\hat{E} \subseteq E'$$

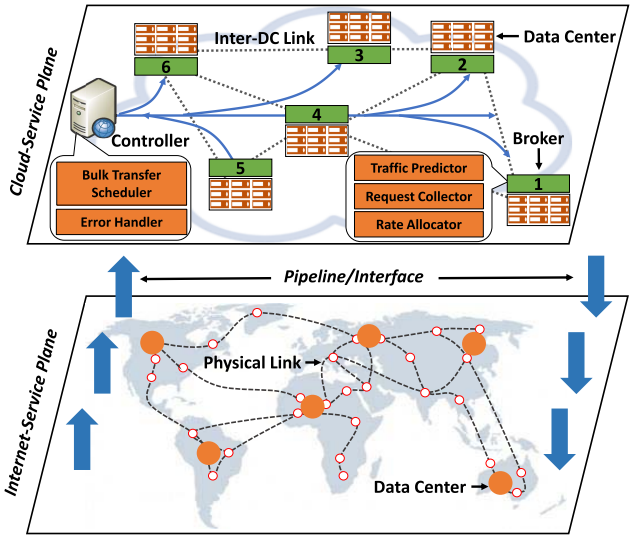



Fig. 3. System overview of Butler.

Then it can be easily proved that

$$\sum_{e\in\hat{E}} u_e \le \sum_{i\in I} U^i_{p^*_i}$$

Thus, the inequality

$$R = \frac{\Omega_1}{\Omega_0} < 1 + \max_{i\in I} \frac{U}{v_i}$$

holds. This completes the proof. ■

Typically, the duration of a billing cycle is one day or one week, i.e., several tens of thousands or hundreds of thousands of seconds [9]. Since the demand of a bulk transfer is in the order of TB or PB [15], the size of $i$ can be hundreds of millions of GBs. The value is potentially up to four orders of magnitude higher than that of the duration of a billing cycle. As $v_i$ would be very large, we expect that PDA could achieve a tight approximation ratio against the optimal solution.

## IV. Butler: A Bulk Transfer Scheduler

Following the architecture of software-defined Inter-DC WANs as shown in Fig. 1, we design a system with PDA to schedule the bulk transfers cost-efficiently in Inter-DC WANs and we call it "**Butler**" (i.e., the abbreviation of **Bu**lk **t**ransfer schedu**ler**).

### A. Overview

Butler is a centralized system to economically schedule the bulk transfers over Inter-DC WANs. As shown in Fig. 3, the functions of Butler are divided into two logical planes: cloud-service plane and Internet-service plane. The cloud-service plane is in charge of the scheduling of bulk transfers and transfers of interactive traffic. The Internet-service plane provides the cloud-service plane with bandwidth and charges. With the facilitation of Butler, data center operators could ignore the implementation details of the Internet-service plane (or Inter-DC WAN), while ISPs just need to operate on the Internet-service plane without considering the detailed contents transferring between data centers.

Butler consists of a *controller* and multiple *brokers*. Each data center is equipped with a broker. It implements a strict bandwidth sharing policy by giving the interactive traffic the highest priority in transmission. The controller maintains the global information about the network, and is responsible for scheduling bulk transfers and handling errors. For fault-tolerance, data center operators could replicate controllers and place them in different data centers, with one elected as the master through Paxos [28]. Since ISPs need to record the bandwidth usages every 5 minutes, also called *scheduling period*, to calculate bandwidth cost [12], brokers in data centers periodically submit the volume of interactive traffic predicted and the bulk transfers requested to the controller to allocate the transmission bandwidth. It is noteworthy that the volume of the interactive traffic can be well predicted for short transmission period such as 5-minute [2], [4].

A broker has three modules: (1) Traffic Predictor (TP), which predicts the volume of interactive traffic in the next scheduling period; (2) Request Collector (RC), which collects the transmission requests; (3) Rate Allocator (RA), which enforces the rate allocation following the instruction of the controller. A controller has two modules: Bulk Transfer Scheduler (BS) and Error Handler (RH), which are responsible for finding the schedule of bulk transfers and handling various errors (e.g., link failures and mispredictions). Butler works as follows: in a typical scheduling period, applications submit bulk transfer requests to RC, which forwards these requests to BS of the controller at the end of this scheduling period. The schedule determined by the controller is sent to RA to execute in the next scheduling period. Considering the latency-sensitive characteristic of interactive traffic, it is predicted and scheduled in brokers, while the information on the traffic volume is forwarded to the controller. BS runs in every time slot to fit the predictable window of interactive traffic. It schedules the bulk transfers with the minimal bandwidth cost, taking into account the required bandwidths of interactive traffic and various network states, e.g., link failures. Any newly arrived bulk transfer can be scheduled to transfer within a scheduling period, and the waiting is negligible compared with its long duration in the unit of hours to days.

### B. Broker

The design details of the modules in brokers are presented as follows:

*1) Traffic Predictor (TP):* Since interactive traffic in Inter-DC WANs is bursty and highly diurnal, it sometimes causes high traffic volumes on links for a few time slots, resulting in high bandwidth cost for data center operators. To handle interactive traffic properly, TP predicts the demand of interactive traffic based on the average usage of interactive services in the last five minutes [2], [4]. To mitigate the loss caused by misprediction, the mispredictions in past time slots are also taken into consideration in the prediction process. For the predicted interactive requests, TP will determine the routing paths and the traffic volumes of all links traveled.

It sends the routing paths for predicted interactive requests and the traffic volume information to the controller every time slot for it to consider when scheduling the bulk transfers. It also sends similar information to RA for it to efficiently schedule interactive requests according to the path and volume information given.

*2) Request Collector (RC):* In a data center, applications submit their Inter-DC bulk transfer requests to RC. Each request contains the necessary information: source, destination, start time, maximum transmission time and transfer demand. We have introduced how Butler schedules the transfer in a single billing cycle. If a request spreads across two or more billing cycles, we divide it into multiple parts based on its start time and deadline. Suppose there exists a request across two billing cycles with the duration of 6 time slots between its start time and its deadline, 2 of which are in the first billing cycle and 4 time slots are in the second billing cycle. We will divide it into two sub-transfers with the ratio of their demands $1:2$. Then we treat each sub-transfer as a single transfer in the two billing cycles. In our implementation, servers send requests to brokers with sockets. Bulk transfer requests will be temporarily stored in RC. At the end of each time slot, RC sends these bulk transfer requests to the controller in batches. Interactive requests will be forwarded to RA instantly.

*3) Rate Allocator (RA):* It enforces the rate allocated by the controller for bulk transfers on physical and virtual machines [29]–[31]. For the interactive requests received, if the volumes are accurately predicted by TP or the actual transmission volumes are smaller than the predicted ones, RA will route them on the paths determined by TP. If the actual volumes are larger than the predicted ones, where mispredictions occur, RA will preferentially transfer data from interactive requests, and delay the bulk transfers that share the same paths but have the deadlines far away (see the details in Section IV-C.1). RA sends network and transfer states to the controller in each time slot.

### C. Controller

We present the design details of the modules in controller as follows:

*1) Bulk Transfer Scheduler (BS):* BS is the key module for bulk transfer scheduling and we deploy PDA in it. Generally, BS receives requests from RC and hands down the rate allocations to RA to enforce. Considering that interactive traffic accounts for 5-15% of the total traffic in Inter-DC WANs and it can be largely predicted for short transmission period such as 5-minute [2], [4], TP runs to predict the volume of interactive traffic in the near future. As services submit bulk transfer requests independently and ISPs record the bandwidth usage in each 5-minute interval, BS runs PDA in each scheduling period (i.e., 5 minutes) to manage the bandwidth usage. TP predicts the volume of interactive traffic that will flow through link $e$ at next scheduling period and add it to the left-hand side of inequality (4) to require bandwidth reservation for the incoming interactive data.

*2) Error Handler (EH):* EH is located inside the controller of Butler, and is responsible for handling various types of errors, including link failures and mispredictions of interactive traffic. Link failure is a typical error in Inter-DC WANs. To prevent the violation of deadlines caused by link failures, we set the deadlines of transfers one scheduling period (i.e., 5 minutes) ahead in our scheduling. Since current techniques such as Dionysus [32] can update the network status (e.g., rate reallocation) in several seconds and the controller calculate the transfer scheduling in each scheduling period, our setting is sufficient for Butler to relieve the bad effects caused by link failures.

Misprediction is inevitable and the unpredicted data burst will preempt the bandwidth allocated to bulk transfers. To mitigate it, we set a small amount of headroom on the basis of the calculated bandwidths to absorb the traffic burst. In the case that the headroom is not enough to handle the burst, RA delays some bulk transfers that share the same paths. It preferentially delays the transfer with the farthest deadline, and repeats the procedure until the unpredicted burst and the bulk transfers with near deadlines can be fully satisfied. If the above procedure does not work, RA employs more bandwidth[1] to guarantee the deadlines of transfers. In practice, a small number of burst will not introduce more bandwidth cost under the popular percentile-based charging scheme [12].

### D. Prototype

We implement a prototype system of our Butler with C++ and Python. We call the advanced integer programming solver, Gurobi Optimizer [34], to determine the bandwidth allocation. In our prototype, the controller collects information from the brokers located in data centers and commands these brokers to control the data rates of transfers. The controller performs the bulk transfer scheduling every 5 minutes. In each data center, we implement the three modules in broker (i.e., TP, RC and RA) with Python. TP also runs every 5 minutes to predict interactive traffic. Then it sends the traffic volumes of interactive requests on links to the controller. RC runs as daemon process to collect transfer requests from applications. It records the bulk transfer requests and sends them to the controller in batch every 5 minutes. When interactive requests arrive, it forwards them to RA immediately. Rate Allocator reports the status of transfers to the controller, and receives schedule commands from the latter. Then it enforces rate allocations with Linux TC. RA also runs as daemon process in each data center. We use socket programming to realize the communications between above modules.

## V. Evaluation

In this section, we conduct extensive evaluations to test the performance of Butler.

### A. Evaluation Methodology

In our simulations, we use the Inter-DC network topology of Google [3], which consists of 12 data centers and 19 bi-directional links (as shown in Fig. 4). We distribute the data

[1]ISPs provide sufficient bandwidth and charge the data center operators based on the actual usages [33].

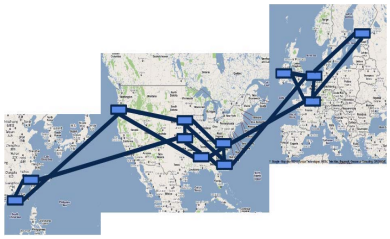


Fig. 4. B4 worldwide deployment [3].

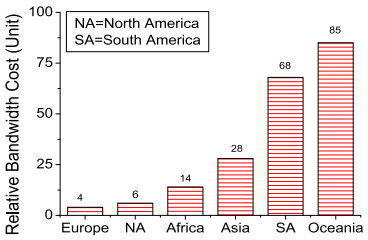



Fig. 5. Relative bandwidth costs.

centers to 6 geographical regions. The bandwidth prices are set based on the relative bandwidth costs in [11]. We show the relative cost of bandwidth in Fig. 5. We set one day (24 hours) as a billing cycle and adopt the popular percentile-based scheme to charge for the bandwidth [9]. We record the traffic volumes of each 5-minute interval, based on which we update the charging bandwidth with the common approach as described in [12]. The unit of bandwidth is set to 10Gbps [10]. To set the bandwidth price properly, we refer to the relative bandwidth cost around the world presented by CloudFlare [11], e.g., the cost of per unit charging bandwidth in Europe is 4 units. The bandwidth cost is calculated according to the outgoing traffic volumes of data centers.

*1) Transfer Generation:* Referring to [4], [9], [15], we generate Inter-DC transfers with a synthetic model as follows. The arrival time of bulk transfer requests follows Poisson distribution model with arrival rate $\lambda$ per timeslot (i.e., 5 minutes). The demand of each bulk transfer follows an exponential distribution with a mean of 5TB in simulation experiments. The source data centers and destination data centers of transfers are selected randomly. Besides bulk transfers, interactive traffic is randomly generated and accounts for 5-15% of the total traffic demand.

*2) Comparing Solutions:* As there are many existing Inter-DC traffic engineering (TE) with different goals and manners, we select the representative ones to compare with Butler. To make a fair comparison, Butler and the comparing solutions require no more storage resources on current systems. We compare the performance of Butler with three TE solutions: (1) *Basic* [22] selects the min-cost[2] routing path to deliver bulk transfers at desired rate, i.e., the quotient of transfer demand divided by the maximum tolerable transfer time. (2) *SWAN* [2] schedules Inter-DC transfers to maximize the network utilization of Inter-DC WAN. (3) *Pretium*[3] [9], which we modify to minimize the bandwidth cost. As Amoeba [4] aims to meet as many deadlines of user requests as possible with fixed bandwidths while our solution focuses on guaranteeing all deadlines with as little bandwidth cost as possible, it is difficult to compare their performance. Thus Amoeba [4] is not compared with ours.

[2]Among all the routing paths for a pair of data centers, the summation of bandwidth prices of links on the min-cost routing path is the least.

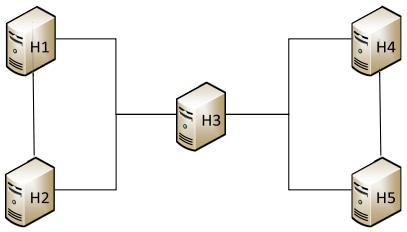


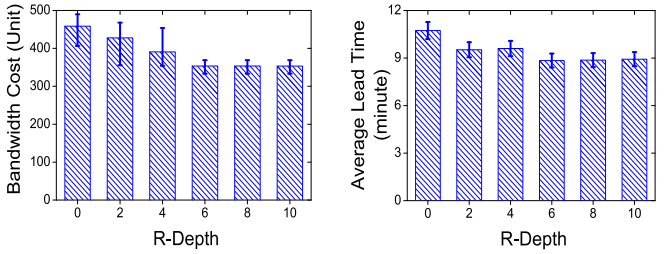

Fig. 6. Basic topology of the testbed.

*3) Testbed Setup:* As shown in the basic topology in Fig. 6, our testbed consists of 5 hosts which act as data centers in different geographical areas and 6 bidirectional links. Each host is connected to others with 1000Mbps links. A host is equipped with Intel Core i5-7500 CPU, 8GB Memory and 1G Ethernet NIC, and runs Ubuntu 16.04 64-bit version with Linux 4.6.2 kernel. In our testbed experiment, the arrival time of bulk transfer requests follows a Poisson distribution with arrival rate $\lambda$ per timeslot. The demand of each bulk transfer follows an exponential distribution with a mean of 5GB. The unit of bandwidth is set as 100 Mbps. We generate the transfers with iperf [35]. We set billing cycle consisting of 50 time slots. The RTT of link is set according to the geographical distances between different areas. In our experiments, we compare the performance of Butler with three comparing solutions, including Basic [22], SWAN [2] and Pretium [9].

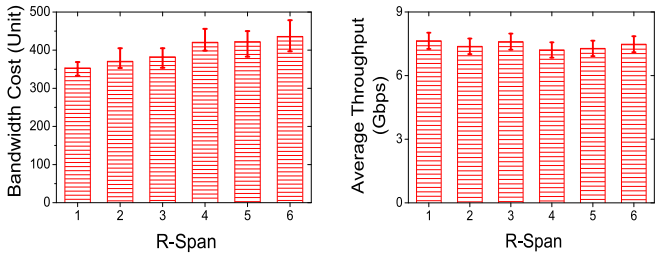
### *B. Performance of PDA*

To begin with, we evaluate different parameter settings of PDA to look for the optimal configurations. In Fig. 7, we evaluate the impact of different R-Depth and R-Span. We first evaluate the impact of R-Depth on bandwidth cost and flow performance, where we fix the value of R-Span to 1. As shown in Fig. 7a, when R-Depth is less than or equal to 6, with its increase, the bandwidth cost drops up to 30.4%. However, when R-Depth exceeds 6, the bandwidth cost remains unchanged. This indicates that PDA has achieved the minimum cost. In Fig. 7b, we evaluate the *lead time* of bulk transfers, which is defined as the deadline of a bulk transfer minus its actual completion time. As the lead time of bulk transfers for Butler is always non-negative, it indicates that Butler can always meet the transfer deadlines, and a larger lead time indicates that the transfer is completed a longer time before the deadline. The trend is similar to that of Fig. 7a. When the R-Depth increases from 0 to 6, the average lead time decreases from 10.7 to 8.8, and it only changes a little when R-Depth increases further.

[3]The original objective of Pretium is to maximize social welfare. It uses the top 10% utilization values of links to approximate the true bandwidth usage.

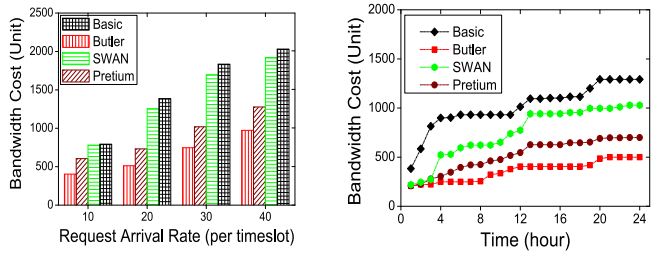



(a) Bandwidth cost over R-Depth (b) Average lead time over R-Depth

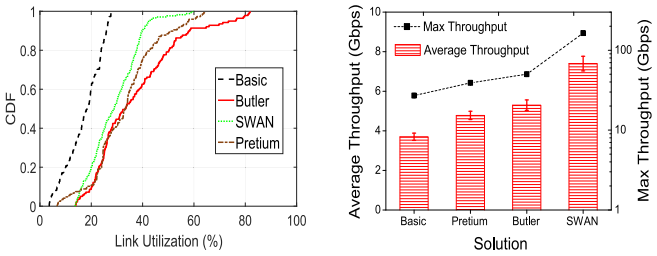



(c) Bandwidth cost over R-Span (d) Average throughput over R-Span

Fig. 7. Performance of PDA.

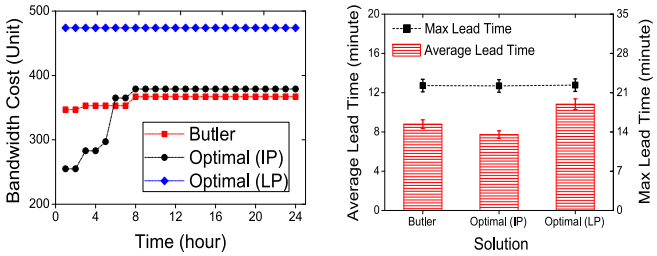



(a) Average bandwidth cost (b) Bandwidth cost variation




(c) CDF of link utilization (d) Throughput comparison

Fig. 8. Butler vs. Inter-DC TE.




(a) Bandwidth cost comparison (b) Lead time comparison

Fig. 9. Butler vs. Optimal Solutions.

We also evaluate the impact of R-Span with different values, with the value of R-Depth fixed to 6 according to the above study. When R-Span increases from 1 to 6, different from R-Depth, the bandwidth cost increases simultaneously. In Fig. 7c, the average bandwidth cost has increased for 23.5% when R-Span changes from 1 to 6. It appears that, rounding one fractional bandwidth at one time is the best strategy for Butler. In Fig. 7d, we see that the average throughput of bulk transfers under different settings of R-Span are almost identical. Thus the change of R-Span has little impact on the throughput of bulk transfers. Combined with Fig. 7a and Fig. 7b, we conclude that adjusting R-Depth and R-Span can reduce the bandwidth cost effectively without a significant impact on the flow performance.

### C. Butler vs. Inter-DC TE

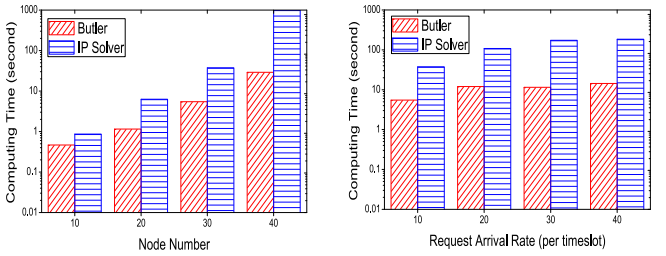
In Fig. 8, we compare Butler with other Inter-DC TE techniques, including Basic [22], SWAN [2] and Pretium [9]. To minimize the total bandwidth cost, Butler lowers the peak bandwidth usage while Pretium lowers the top 10% utilization values of links [9]. To make the results comparable, we let all schemes guarantee the deadlines of bulk transfers. In Fig. 8a, we compare the bandwidth costs of these schemes under different request arrival rates. With the increase of request arrival rate, the bandwidth costs of all schemes increase. However, Butler has a much slower growth compared with other schemes. Since Butler adopts our well-designed rounding strategy to directly optimize the bandwidth, its advantage on reducing the bandwidth cost becomes larger as the request arrival rate increases. When bulk transfers arrive at the rate of 40 per time slot, the bandwidth costs of Pretium, SWAN and Basic are 61%, 97.4% and 108.7% higher than that of Butler, respectively. As shown in Fig. 8b, the bandwidth costs of all schemes increase over time. However, the bandwidth cost of Butler increases slower than others. At the end of running, the bandwidth costs of the four solutions are 500, 699, 1029 and 1291 units, where the cost of Butler is up to 62% lower. In Fig. 8c, the average link utilization of Butler is 2.16x, 1.27x and 1.12x that of Basic, SWAN and Pretium, as Butler attempts to reduce the overall bandwidth cost by making full use of the purchased bandwidth. In Fig. 8d, SWAN has the highest average and maximum throughput as it aims to improve the throughput of Inter-DC flows. Basic transfers at the desired rates, and its throughput is the least among all solutions, no matter the average throughput or the maximum throughput. Making full use of the purchased bandwidth as shown in Fig. 8c, Butler achieves higher average and maximum throughput than those of Pretium.

### D. Butler vs. Optimal Solutions

In Fig. 9, we compare the scheduling of Butler with the optimal solutions of CESBT and Relaxed-CESBT, which are denoted as Optimal (IP) and Optimal (LP) respectively. Optimal (LP) rounds up the fractional charging volumes in the optimal solution of Relaxed-CESBT. We show the bandwidth cost variations of these schemes in Fig. 9a. At the beginning of the billing cycle, the bandwidth cost of Optimal (IP) is lower than that of others. With the arrivals of more bulk transfer requests, Butler schedules them in a more economic way,

TABLE III
BANDWIDTH COSTS (BASELINE = 1)

| Error Type | | Link Failure | | | | Misprediction | | | |
|---|---|---|---|---|---|---|---|---|---|
| Error Ratio | | 10% | 20% | 30% | 40% | 10% | 20% | 30% | 40% |
| Req. Arr. Rate (per timeslot) | 10 | 1.002 | 1.116 | 1.135 | 1.162 | 1.042 | 1.064 | 1.067 | 1.090 |
| | 20 | 1.002 | 1.114 | 1.196 | 1.177 | 1.022 | 1.027 | 1.034 | 1.076 |
| | 30 | 1.008 | 1.118 | 1.126 | 1.151 | 1.007 | 1.042 | 1.042 | 1.043 |
| | 40 | 1.001 | 1.114 | 1.131 | 1.210 | 1.007 | 1.012 | 1.034 | 1.042 |




(a) Computing time with different network scales

(b) Computing time with different request arrival rates

Fig. 10. Scalability of Butler.

thus its bandwidth cost increases much slower than those of Optimal (IP) and Optimal (LP). In each time slot, Optimal (IP) tries to minimize the bandwidth cost, which sacrifices its capacity of accommodating future requests. At the end of the billing cycle, the bandwidth cost of Butler is slightly lower than that of Optimal (IP), and is up to 22.6% lower than that of Optimal (LP). This is because PDA uses effective rounding strategies, while Optimal (LP) simply rounds up all fractional traffic volumes, which leads to a higher bandwidth cost. In Fig. 9b, there exist subtle differences between the maximum lead time of different solutions. However, for the average lead time of bulk transfers, Optimal (LP) outperforms Butler and Optimal (IP), as Optimal (LP) benefits from the use of more link capacity at the cost of higher usage charge.

### *E. Scalability*

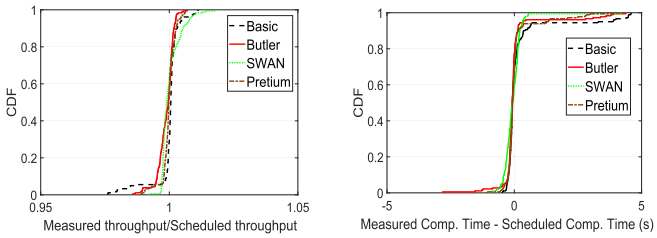
To demonstrate the scalability of Butler, we compare it with Gurobi Optimizer [34] using large-scale simulations. We measure the computing time of Butler and IP Solver under different network scales and request arrival rates. Our measurements are conducted in a desktop, which is equipped with Intel Core i5-7500 3.40GHz CPU, 8GB Memory, 240GB SSD and runs Windows 10 64-bit version. In Fig. 10a, we fix the request arrival rate to 10 per time slot. The network topologies are randomly generated connected networks, and the number of links increases with the number of node. We measure the computing time of Butler and IP Solver using the same workload. When the number of nodes grows, the computing time of IP Solver increases exponentially. When the network topology has 40 nodes, its computing time is longer than 1000 seconds, which is intolerable for data center operators. However, the computing time of Butler has a slow growth. When there are 40 nodes in the network, the computing time of Butler is less than 100 seconds. We evaluate the impact of request arrival rate on the computing time of different schemes. We fix the network topology with 30 nodes. When the request arrival rate increases, the computing time of IP Solver increases much faster than Butler. For example, when the request arrival rate is 40 per time slot, the computing time of IP Solver is 12.9x that of Butler. With the proliferation of cloud services, Inter-DC WANs could benefit from the high scalability of Butler on performing cost-efficient scheduling of bulk transfers.

### *F. Robustness*

To evaluate the robustness of Butler under different levels of link failures and mispredictions, we set different error ratios (ERs) and evaluate the bandwidth cost of Butler. In Butler, the deadline of bulk transfer is set to one time slot earlier than the actual deadline and the headroom for mitigating misprediction is set based on the volume of the maximum burst in the previous time slots. We set some links as "failed" to simulate link failures, but guarantee the full-connectivity of the remaining network. We manually change the size of generated interactive transfers to simulate the misprediction errors. The results are presented in Table III. For a certain request arrival rate, we set the bandwidth cost as 1.0 when there is no error (i.e., ER $= 0$). We present the relative bandwidth costs under different ER and error types (ETs). With the increase of ER, handling errors incurs higher bandwidth cost, while the increase in the cost is always less than 21% in our simulations. It indicates that Butler can efficiently handle the errors during scheduling. Under the same settings of ER, handling link failures incurs more bandwidth costs than that of handling mispredictions in most cases. This is because link failures will force the controller to select new routing paths for the affected transfers, while mispredictions can be mitigated or eliminated by the headroom and periodic calculations of the controller. From the above results, we conclude that Butler is a robust solution for cost-efficient scheduling of bulk transfers.

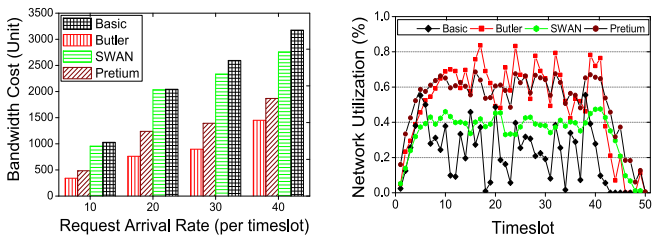
### *G. Testbed Experiments*

In this subsection, we first measure the accuracy of our enforcement. Then we measure the bandwidth cost under different request arrival rates and the network utilization of each time slot for different solutions.




(a) Throughput comparison (b) Completion time comparison

Fig. 11. Deviation between schedules and testbed results.




(a) Bandwidth cost (b) Network utilization ($\lambda = 40$)

Fig. 12. Testbed experiments.

In Fig. 11a, the ratio of measured throughput to the scheduled throughput is between 0.95 and 1.05. It shows that the measured throughput matches the scheduled ones. In Fig. 11b, for all bulk transfers, the difference between the scheduled completion time and measured completion time is less than 5 seconds, which is negligible compared with the long duration of a time slot in the unit of minutes. We infer that, these differences are caused by iperf, which copies data between user space and kernel space, and TC, which needs time to perform rate limitation on NICs.

Fig. 12a shows the similar changing trend as shown in Fig. 8a. As the request arrival rate increases, the bandwidth costs of all solutions increase. Good rounding strategy and direct optimization of peak bandwidth usage help Butler achieve the lowest bandwidth cost among the four solutions. In Fig. 12b, we measure the network utilization of each time slot when the request arrival rate is 40 per time slot. Benefited from its cost-efficient scheduling strategy, Butler has a higher network utilization than others in most time slots, which increases the average network utilization 1.4x. The result shows that Butler and Pretium make full use of the purchased bandwidth to reduce the overall bandwidth cost and achieve similar network utilization. Despite that SWAN attempts to improve the throughput and network link utilization, to guarantee the deadlines of transfers, its network utilization is affected and lowered. SWAN and Basic schedule bulk transfers without considering the delivery cost, thus they have higher bandwidth costs than Butler and Pretium.

## VI. Related Work

There are many recent efforts on Inter-DC traffic engineering and bulk transfer scheduling in Inter-DC WANs, but none of them can solve CESBT directly.

Benefitted from the emerging SDN technologies, SWAN [2] and B4 [3], [7] drive links in Inter-DC WANs to near 100% utilization and balance the capacity against application priority/demands. Tempus [16] appropriately packs long-lived transfers across network paths and future time steps. It maximizes the minimal delivered fraction of transfers before deadlines and achieves the fairness among transfers. These solutions work with fixed bandwidth. They neither guarantee service deadline nor consider the transmission cost of Inter-DC WANs. In contrast, Butler takes into account the bandwidth costs and schedules bulk transfers with guaranteed deadlines.

Amoeba [4] takes one step further by accommodating more user requests with guaranteed-deadlines under the limits of bandwidth. Similar to SWAN [2] and B4 [3], it has not considered the bandwidth cost incurred by Inter-DC transfers [36]. It accepts more user requests to fully utilize the fixed bandwidth while Butler considers the transmission cost and aims to serve requests with the minimal bandwidth cost.

There are some work using pricing schemes to optimize the cost-efficiency of Inter-DC WANs. For example, Pretium [9] combines dynamic pricing and TE to maximize the social welfare. Tenants sometimes have to adjust their traffic and compromise their service quality to not violate the payment agreement with the cloud providers. However, Butler schedules Inter-DC transfers without requiring complicated pricing scheme to lower bandwidth cost and guaranteeing deadlines.

OWAN [15] schedules the bulk transfers in Inter-DC WANs by dynamically changing the network-layer topology with optical devices reconfigured. It provides best-effort delivery service but can not guarantee the deadlines of bulk transfers, while Butler performs cost-efficient scheduling under the condition of meeting transfer deadlines.

Many solutions attempt to make full use of the purchased bandwidths in Inter-DC WANs. Metis [6] is a framework that enables cloud providers to earn more service profit, i.e., service revenue minus service cost, in geo-distributed clouds by declining some user requests and efficiently scheduling the accepted ones. It can benefit from scheduling accepted user requests with our algorithm. NetStitcher [8] and Postcard [22] use the store-and-forward approach in the scheduling of bulk transfers. They delay the large data transfers and send them at non-peak hours to fully utilize the leftover bandwidth. They require data centers to have large space to temporarily store bulk data. Different from these schemes, Butler aims to achieve cost-efficient scheduling of bulk transfers with guaranteed deadlines and minimum bandwidth cost, while not requiring extra storage space.

## VII. Conclusion

In this paper, we focus on the cost-efficient scheduling of bulk transfers (CESBT) in Inter-DC WANs while guaranteeing their deadlines. Different from the previous work that may degrade service performance or introduce extra overhead to achieve high network utilization, we take into account the variation of bandwidth price and the deadline requirements of services in our problem formulation. We formulate the problem of CESBT and prove its NP-hardness. To solve it with a guaranteed approximation ratio in polynomial time, we propose the progressively-descending algorithm (PDA) based

on relax-and-round and prove its approximation ratio. We apply PDA in the bulk transfer scheduler, Butler, and build a testbed to evaluate the effectiveness of our scheme. Extensive simulation and testbed experiments demonstrate the advantage of our scheme in significantly cutting down the bandwidth cost while guaranteeing bulk transfers to meet their deadlines.

## REFERENCES


[1] S. Liu and B. Li, "Stemflow: Software-defined inter-datacenter overlay as a service," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2563–2573, Nov. 2017.

[2] C.-Y. Hong *et al.*, "Achieving high utilization with software-driven WAN," in *Proc. ACM SIGCOMM*, 2013, pp. 15–26.

[3] S. Jain *et al.*, "B4: Experience with a globally-deployed software defined WAN," in *Proc. ACM SIGCOMM*, 2013, pp. 3–14.

[4] H. Zhang *et al.*, "Guaranteeing deadlines for inter-datacenter transfers," in *Proc. ACM EuroSys*, 2015, Art. no. 20.

[5] A. Roy, H. Zeng, J. Bagga, G. Porter, and A. C. Snoeren, "Inside the social network's (datacenter) network," in *Proc. ACM SIGCOMM*, 2015, pp. 123–137.

[6] Z. Yang *et al.*, "Towards maximal service profit in geo-distributed clouds," in *Proc. IEEE ICDCS*, Jul. 2019, pp. 442–452.

[7] C.-Y. Hong *et al.*, "B4 and after: Managing hierarchy, partitioning, and asymmetry for availability and scale in google's software-defined WAN," in *Proc. ACM SIGCOMM*, 2018, pp. 74–87.

[8] N. Laoutaris, M. Sirivianos, X. Yang, and P. Rodriguez, "Inter-datacenter bulk transfers with netstitcher," in *Proc. ACM SIGCOMM*, 2011, pp. 74–85.

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

[10] I. Takanori, "Large-capacity optical transmission technologies supporting the optical submarine cable system," *NEC Tech. J.*, vol. 5, no. 1, pp. 8–12, Feb. 2010.

[11] R. Nitin. *Bandwidth Costs Around the World*. Accessed: Aug. 17, 2016. [Online]. Available: https://blog.cloudflare.com/bandwidth-costs-around-the-world/

[12] D. K. Goldenberg, L. Qiuy, H. Xie, Y. R. Yang, and Y. Zhang, "Optimizing cost and performance for multihoming," in *Proc. ACM SIGCOMM*, 2004, pp. 1–14.

[13] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: Research problems in data center networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 68–73, 2009.

[14] W. Li *et al.*, "Cost-minimizing bandwidth guarantee for inter-datacenter traffic," *IEEE Trans. Cloud Comput.*, vol. 7, no. 2, pp. 483–494, Jun. 2016.

[15] X. Jin *et al.*, "Optimizing bulk transfers with software-defined optical WAN," in *Proc. ACM SIGCOMM*, 2016, pp. 87–100.

[16] S. Kandula, I. Menache, R. Schwartz, and S. R. Babbula, "Calendaring for wide area networks," in *Proc. ACM SIGCOMM*, 2014, pp. 515–526.

[17] Y. Zhang *et al.*, "BDS: A centralized near-optimal overlay network for inter-datacenter data replication," in *Proc. ACM EuroSys*, 2018, Art. no. 10.

[18] Y. Lin, H. Shen, and L. Chen, "EcoFlow: An economical and deadline-driven inter-datacenter video flow scheduling system," in *Proc. ACM Int. Conf. Multimedia*, 2015, pp. 736–737.

[19] H. H. Liu, Y. Wang, Y. R. Yang, H. Wang, and C. Tian, "Optimizing cost and performance for content multihoming," in *Proc. ACM SIGCOMM*, 2012, pp. 371–382.

[20] M. Ghasemi, P. Kanuparthy, A. Mansy, T. Benson, and J. Rexford, "Performance characterization of a commercial video streaming service," in *Proc. ACM IMC*, 2016, pp. 499–511.

[21] P. Mattew. *The Relative Cost of Bandwidth Around the World*. Accessed: Aug. 26, 2014. [Online]. Available: https://blog.cloudflare.com/the-relative-cost-of-bandwidth-around-the-world/

[22] Y. Feng, B. Li, and B. Li, "Postcard: Minimizing costs on inter-datacenter traffic with store-and-forward," in *Proc. IEEE ICDCSW*, Jun. 2012, pp. 43–50.

[23] Z. Yang, Y. Cui, B. Li, Y. Liu, and Y. Xu, "Software-defined wide area network (SD-WAN): Architecture, advances and opportunities," in *Proc. IEEE ICCCN*, Jul. 2019.

[24] Z. Zhang *et al.*, "Optimizing cost and performance in online service provider networks," in *Proc. USENIX NSDI*, 2010, pp. 1–15.

[25] H. Shi *et al.*, "STMS: Improving MPTCP throughput under heterogeneous networks," in *Proc. USENIX ATC*, 2018, pp. 719–730.

[26] C. Raiciu *et al.*, "How hard can it be? Designing and implementing a deployable multipath TCP," in *Proc. USENIX NSDI*, 2012, pp. 399–412.

[27] D. Bienstock, S. Chopra, O. Günlük, and C.-Y. Tsai, "Minimum cost capacity installation for multicommodity network flows," *Math. Program.*, vol. 81, no. 2, pp. 177–199, 1998.

[28] L. Lamport, "The part-time parliament," *ACM Trans. Comput. Syst.*, vol. 16, no. 2, pp. 133–169, 1998.

[29] M. Alizadeh *et al.*, "pFabric: Minimal near-optimal datacenter transport," in *Proc. ACM SIGCOMM*, 2013, pp. 435–446.

[30] K. Nagaraj *et al.*, "NUMFabric: Fast and flexible bandwidth allocation in datacenters," in *Proc. ACM SIGCOMM*, 2016, pp. 188–201.

[31] K. He *et al.*, "AC/DC TCP: Virtual congestion control enforcement for datacenter networks," in *Proc. ACM SIGCOMM*, 2016, pp. 244–257.

[32] X. Jin *et al.*, "Dynamic scheduling of network updates," in *Proc. ACM SIGCOMM*, 2014, pp. 539–550.

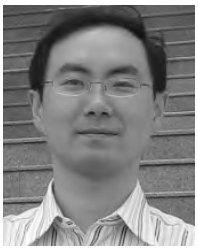
[33] *95th Percentile Bandwidth Metering Explained and Analyzed*. [Online]. Available: https://www.semaphore.com/

[34] *Gurobi Optimizer*. [Online]. Available: http://www.gurobi.com/

[35] *Iperf*. [Online]. Available: https://iperf.fr/

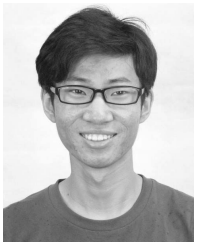
[36] W. Li, X. Zhou, K. Li, H. Qi, and D. Guo, "More peak, less differentiation: Towards a pricing-aware online control framework for inter-datacenter transfers," in *Proc. IEEE ICDCS*, Jun. 2017, pp. 2105–2110.


**Zhenjie Yang** received the B.E. degree in networking engineering from the Dalian University of Technology, Liaoning, China, in 2015. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology, Tsinghua University, Beijing, China. His research interests include data center networking and cloud computing.



**Yong Cui** received the B.E. and Ph.D. degrees in computer science and engineering from Tsinghua University, China, in 1999 and 2004, respectively. He is currently a Full Professor with the Computer Science Department, Tsinghua University. He has published over 100 papers in the refereed conferences and journals with several best paper awards. He has coauthored seven Internet standard documents (RFC) for his proposal on IPv6 technologies. His major research interests include mobile cloud computing and network architecture. He served or serves on the editorial boards of the IEEE TPDS, IEEE TCC, and the *IEEE Internet Computing*. He is currently the Working Group Co-Chair in the IETF.

**Xin Wang** received the B.S. and M.S. degrees in telecommunications engineering and wireless communications engineering from the Beijing University of Posts and Telecommunications, Beijing, China, and the Ph.D. degree in electrical and computer engineering from Columbia University, New York, NY, USA. She is currently an Associate Professor with the Department of Electrical and Computer Engineering, State University of New York at Stony Brook, Stony Brook, NY, USA. Before joining the State University of New York at Stony Brook, she was a Member of Technical Staff in mobile and wireless networking at Bell Labs Research, Lucent Technologies, NJ, USA, and an Assistant Professor with the Department of Computer Science and Engineering, State University of New York at Buffalo, Buffalo, NY, USA. Her research interests include algorithm and protocol design in wireless networks and communications, mobile and distributed computing, as well as networked sensing and detection. She has served in the executive committee and technical committee of numerous conferences and funding review panels, and serves as an Associate Editor for the IEEE TRANSACTIONS ON MOBILE COMPUTING. She received the NSF Career Award in 2005 and the ONR Challenge Award in 2010.



**Yadong Liu** received the B.E. degree in software engineering from the Beijing Institute of Technology, Beijing, China, in 2017. He is currently pursuing the M.S. degree with the Department of Computer Science and Technology, Tsinghua University, Beijing. His research interests are in the areas of data center networking.

**Minming Li** received the Ph.D. degree. He is currently an Associate Professor with the Department of Computer Science, City University of Hong Kong. His research interests include algorithms design and analysis, combinatorial optimization, scheduling, key management, and algorithmic game theory. He is the winner of the City University of Hong Kong Teaching Excellence Award during 2011–2012.

**Shihan Xiao** received the B.Eng. degree in electronic and information engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2012, and the Ph.D. degree from the Department of Computer Science and Technology, Tsinghua University, China. He is currently a Senior Engineer with Huawei 2012 NetLab. His research interests include machine learning in networking, data center networking, and cloud computing.

**Chuming Li** received the B.Eng. degree in network engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2017. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology, Tsinghua University, Beijing, China. His research interests include data center networking and cloud computing.