More Peak, Less Differentiation: Towards A Pricing-aware Online Control Framework for Inter-Datacenter Transfers

Wenxin Li*, Xiaobo Zhou†, Keqiu Li*, Heng Qi*‡, Deke Guo§
*School of Computer Science and Technology, Dalian University of Technology, China.
†Tianjin Key Laboratory of Advanced Networking, School of Computer Science and Technology, Tianjin University, China.
‡Graduate School of Information Science, Nagoya University, Japan.
§College of Information Systems and Management, National University of Defense Technology, China.
Xiaobo Zhou is the corresponding author: xiaobo.zhou@tju.edu.cn

***Abstract*—The emerging deployment of geographically distributed data centers (DCs) incurs a significant amount of data transfers over the Internet. Such transfers are typically charged by Internet Service Providers (ISPs) with the widely adopted *q-th percentile charging model*. In such charging model, the time slots with top $(100-q)$ percent of data transmission do not affect the total transmission cost, and can be viewed as *"free"*. This brings the opportunity to optimize the scheduling of inter-DC transfers to minimize the entire transmission cost. However, very little work has been done to exploit those *"free"* time slots for scheduling inter-DC transfers. The crux is that existing work either lacks a mechanism to accumulate traffic to *"free"* time slots, or inevitably relies on prior knowledge of traffic arrival patterns. In this paper, we attempt to exploit those *"free"* time slots by leveraging diverse time-sensitivities among inter-DC transfers, so as to reduce or even minimize the transmission cost. Specifically, we advocate that a simple principle should be followed: *more traffic peaks* should be scheduled in *"free"* time slots, while *less traffic differentiation* should be maintained among the remaining time slots. To this end, we take advantage of the *Lyapunov Optimization* techniques to design a pricing-aware control framework. This framework efficiently makes online decisions for inter-DC transfers without requiring a prior knowledge of traffic arrivals. To verify our proposed framework, we conduct small-scale testbed implementation. The results show that our framework can realistically reduce the transmission cost by up to 19.38%.**

I. Introduction

Large-scale organizations, such as Google, Microsoft, and Amazon, have made huge investments in building geo-distributed data centers (DCs) to deliver their online services [1, 2]. A key feature of these services is that they continuously produce large volumes of data transmission among different DCs [3]. A recent survey [4] highlighted that 70% of the IT firms have huge data transmission among DCs, ranging from 1Gbps to 10Gbps, nearly half having 5Gbps or more — i.e., from 330 TB to 3.3 PB a month. Such huge data transmission incurs substantial cost for the service provider. In fact, the annual transmission cost is of up to hundreds of millions of dollars, which approximately equals to the power cost of DCs [5]. From the perspective of service provider, the fundamental objective is to reduce, or even minimize the transmission cost incurred by the inter-DC transfers.

Service providers typically purchase bandwidth from Internet Service Providers (ISPs) for their inter-DC transfers, while ISPs charge service providers based on the widely adopted *q-th percentile charging model* [6, 7]. Such charging model can be described as follows: In a charging period of N time slots, the ISP samples the bandwidth usage that a service provider consumed in every time slot and sorts them in ascending order (each time slot is typically 5 minutes). Then, the q-th percentile of all samples is taken as the billed bandwidth. For example, if 95-th percentile charging is in use and the charging period is 30 days, then the billed bandwidth exactly equals to the bandwidth usage of the 8208-th sorted time slot ($95\%\times30\times24\times60/5=8208$). Clearly, in such *$q$-th percentile charging model*, the time slots with top $(100-q)$ percent of data transmission actually do not affect the total transmission cost, and can be viewed as *"free"*. This provides an opportunity to reduce service provider's transmission cost by carefully scheduling the inter-DC transfers.

Further, the diverse time-sensitivities exhibited by different inter-DC transfers also motivate the design of new scheduling methods. For examples, *interactive transfers* are most sensitive to delay, *larger transfers* require to be done within several hours, while *background transfers* are without strict time requirements [1–3, 8]. Hence, we believe that, the transmission cost of a service provider can be effectively reduced, if inter-DC traffic can be accumulated to those *"free"* time slots as more as possible, while satisfying the deadline requirement of each traffic. In such a case, the optimal solution would be to schedule all *"free"* time slots as *traffic peaks*, and at the same time maintain *no traffic differentiation* among the remaining time slots.

Intuitively, it may be a step towards the right direction to design an offline inter-DC transfer scheduling method to obtain the optimal solution. However, such offline optimization inevitably relies on *prior* knowledge of traffic arrival patterns, which are actually unavailable in practice. To the best of our knowledge, no existing methods are in place to exploit

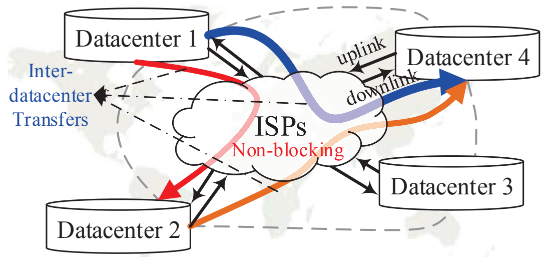


Fig. 1. An illustrative example of inter-DC network model.

the *"free"* time slots in the *q-th percentile charging model* to minimize the transmission cost as well as to guarantee deadlines for inter-DC transfers. *First*, state-of-the-art methods on inter-DC traffic either lack a mechanism to accumulate traffic to *"free"* time slots [8, 9], or cannot provide deadline guarantees for inter-DC transfers [10]. *Second*, although some Internet traffic scheduling methods investigated the impact of the percentile charging model, they either require prior knowledge of future traffic demand [7], or assume uniform deadline requirements for all traffic [11].

In this paper, we attempt to minimize the transmission cost of inter-DC transfers by fully exploiting the advantages of *"free"* time slots in *q-th percentile charging model* and the diverse deadline requirements among inter-DC transfers. To this end, we design a pricing-aware online control framework to practically schedule inter-DC transfers, without prior knowledge of the traffic arrival patterns. Specifically, we formulate a stochastic optimization problem, which takes into account different percentile values across DCs, practical constraints of heterogeneous link capabilities, and different time-sensitivities of inter-DC transfers. Nevertheless, it is impractical to obtain an optimal solution for this problem, due to the unknown information of future traffic arrivals. Thus, we are motivated to transform it into a relaxed problem, which is then solved by designing an online control algorithm based on *Lyapunov Optimization* techniques [12, 13]. We evaluate our framework using small-scale testbed implementation. The results show that our framework reduces the transmission cost by up to 19.38%, compared to the default fair sharing method.

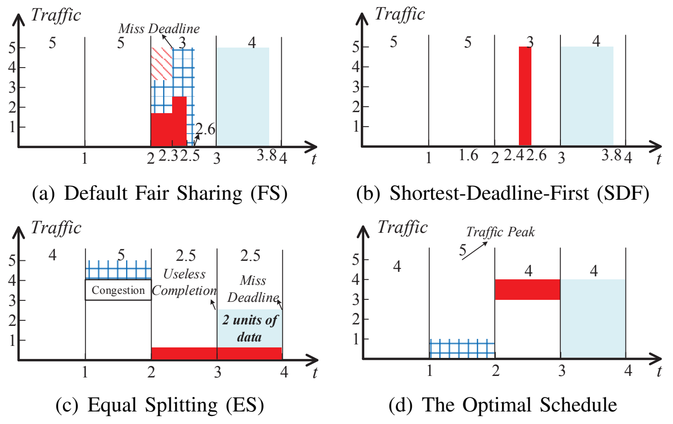
The rest of this paper is organized as follows. Section II introduces the background and motivation and Section III presents the system model and problem formulation. In Section IV, we describe the pricing aware online control framework. In Section V, we evaluate and analyze the performance of our proposed framework. We summarize the related work in Section VI and conclude in Section VII.

II. Background and Motivation

In this section, we present the inter-DC network model and the advantages of pricing-aware scheduling.

A. Inter-DC network model

In this paper, we consider an inter-DC network model where all DCs, geographically distributed across the world, connect to the ISPs with dedicated uplinks and downlinks, as shown in Fig. 1. To complete an inter-DC transfer, each DC only needs to send data to ISPs, and then the ISPs forward the data to the destination DC. Many recent studies [14, 15] have revealed that the ISPs' networks are non-blocking, and the network bottlenecks only appear between DCs and ISPs. Such an inter-DC network model is widely used in practice, and can ease the presentation and analysis due to omit the routing details inside the ISP networks. Some routing proposals [9, 10] are orthogonal to our methods in this paper, and can further reduce the transmission cost of inter-DC transfers if combined with our work.



(a) Default Fair Sharing (FS) (b) Shortest-Deadline-First (SDF) (c) Equal Splitting (ES) (d) The Optimal Schedule

Fig. 2. The billed bandwidth usage under the 75-*th percentile charging model* for (a) FS is 5; (b) SDF is 5; (C) ES is 4; (d) optimal schedule is 4. Both FS and ES miss some flows' deadlines, while SDF and optimal schedule accommodates all flows with deadlines guaranteed. The bandwidth consumption on each time slot is listed in the top of each sub-figure.

B. Potential benefits of pricing-aware scheduling

Instead of directly scheduling inter-DC transfers regardless of the *percentile charging model*, pricing-aware scheduling efficiently utilizes the *"free"* time slots for data transmission, and thus significantly reduce the transmission cost.

Consider an example as shown in Fig. 2, a link is capable of serving up to 5 units of data in one time slot, while carries four flows in a charging period of 4 time slots. The data sizes of the four flows are $8, 4, 1, 4$, respectively, while the deadlines are the end of the 2-th, 3-th, 3-th, 4-th time slot, respectively. Assuming flows arrive at the link at the beginning of each time slot and the 75-th percentile charging model is in use (second largest in this case), the billed bandwidth usage for four scheduling methods are illustrated in Fig. 2. The default fair sharing ensures fairness among concurrent flows in a link, which misses one flow's deadline and incurs high billed bandwidth usage [16]. Shortest-Deadline-First follows the shortest- or smallest-first policy [17, 18]. It is effective in guaranteeing deadlines for the four flows, but results in high billed bandwidth usage. Recently proposed equal splitting schedule mainly considers that all flows can be delayed by a uniform time D, e.g., one time slot, and splits the arriving flows into $D{+}1$ sets of equal size [11]. Then each set is served in one time slot. It can reduce the billed bandwidth usage with large value of D, but may lead to high deadline miss rate as well as significant congestion in some time slots. Finally, the optimal schedule makes one time slot become the traffic peak, and maintains no traffic differentiation among the other three time slots. The time slot of traffic peak is what we call the *"free"* time slot, and it does not affect the total cost based on the *percentile charging model*.

III. System Model and Problem Formulation

A. System model

We consider an inter-DC network where a service provider runs its service over a set of DCs, $\mathcal{M}=\{d_1, d_2, \cdots, d_M\}$. For each DC d_j, let U_j^E and U_j^I denote the bandwidth capacities of its uplink l_j^E and downlink l_j^I, respectively. We consider a system that operates in a discrete-time mode, where the time can be divided into T ($T\in\mathbb{N}^+$) time slots. Each time slot t ($=0, 1, \cdots, T-1$) has a same duration, e.g., 5 minutes. Let S denote the total number of available sessions. At time t, session s transmits exactly one flow at a rate of $r_s(t)$, with the remaining data size and time till deadline being denoted as $\lambda_s(t)$ and $\tau_s(t)$, respectively. Actually, such deadline information can be passed to the transport layer [19]. In such case, let $e_s(t)=\lambda_s(t)/\tau_s(t)$ denote the expected rate for session s at t. Let $x_j^E(t)$ denote the aggregate bandwidth usage on the uplink l_j^E of d_j at t. It is calculated as $x_j^E(t)=\sum_{s\in S(l_j^E)} r_s(t)$, where $S(l_j^E)$ is the set of flows that pass through the uplink l_j^E. Similarly, we have $x_j^I(t)=\sum_{s\in S(l_j^I)} r_s(t)$ for the downlink l_j^I of d_j at t. For each d_j, let q_j denote the percentile that it complies to, let c_j and B_j denote the per unit bandwidth cost and the committed bandwidth, respectively. Consider that each charging period consists of N time slots, emerging $K=T/N$ charging periods. At the end of a charging period k ($=0, 1, \cdots, K-1$), the percentile billed bandwidth for uplink l_j^E can now be calculated as follows:

$$b_j^E(k) = P_{q_j}(x_j^E(kN), \cdots, x_j^E(kN+N-1)), \qquad (1)$$

where $P_{q_j}(\cdot)$ is a function defined as the $\lceil \frac{100-q_j}{100} N \rceil$-th largest number in the bandwidth usage sequence of a charging period. Similarly, the percentile billed bandwidth for the downlink l_j^I at charging period k is calculated as:

$$b_j^I(k) = P_{q_j}(x_j^I(kN), \cdots, x_j^I(kN+N-1)). \qquad (2)$$

Finally, the actual bandwidth that the service provider needs to pay for its DC d_j during charging period k is defined as

$$b_j(k) = \max\{b_j^E(k), b_j^I(k), B_j\}. \qquad (3)$$

Given this definition, we actually need to schedule traffic to *"free"* time slots as much as possible, and simultaneously utilize the committed bandwidth at other time slots with best efforts, so as to follow the simple principle of *"more peak, less differentiation"*.

B. Problem formulation

Given the above model, we now formulate the inter-DC transfer scheduling problem as a stochastic optimization **P1** that minimizes the transmission cost, and at the same time guarantees deadlines for inter-DC transfers.

$$\min_{r_s(t)} \quad \overline{C} = \lim_{T\to\infty} \frac{1}{\frac{T}{N}} \sum_{k=0}^{\frac{T}{N}-1} \sum_{j=1}^{M} c_j b_j(k) \qquad (4)$$

$$\text{s.t.} \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} (e_s(t) - r_s(t)) \le 0, \forall s, \qquad (5)$$

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{s\in S(l_j^E)} r_s(t) \le U_j^E, \forall j, \qquad (6)$$

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{s\in S(l_j^I)} r_s(t) \le U_j^I, \forall j, \qquad (7)$$

$$r_s(t) \ge 0, \forall s, \forall t. \qquad (8)$$

Clearly, Eq. (4) is the objective function that minimizes the *long term average of transmission cost* across all DCs. Eq. (5) ensures that for every flow that requires $e_s(t)$, the allocated rate $r_s(t)$ is larger than $e_s(t)$ on average. This constraint is an approximation to the realistic inter-DC traffic, where the flows cannot be infinitely long. By incorporating this constraint, we are essentially guarantee the deadlines for inter-DC transfers. Eq. (6) (or Eq. (7)) enforces the long-term average bandwidth usage not to exceed the uplink (or downlink) bandwidth capacity for each DC. Here, the basic rationale is that the long-term average traffic load generated in the inter-DC network can be handled by the network capacity, such that the network can be stabilized with proper flow scheduling scheme. Otherwise, no mechanism can do to avoid packet loss. Eq. (8) is simply the non-negativity constraint for the decision variables $r_s(t)$.

Problem **P1** is a long-term optimization problem, where the current control decisions are coupled with the future decisions. For example, current decisions on inter-DC flow scheduling may delay excessive flows and hence block the transfer of future flows. To solve such long-term optimization, one may design an offline optimal scheduling algorithm, or leverage the dynamic programming techniques [20]. However, it encounters two challenges when it comes to realistic networks: *(1)* The traffic arrival pattern is usually unknown in advance, yet is difficult to be accurately predicted; *(2)* The percentile function $P_{q_j}(\cdot)$ relies on the bandwidth usage over a large number of time slots in a charging period. These challenges make it infeasible to identify and use the *"free"* time slots, and thereby impractical to reduce or minimize the transmission cost.

IV. Pricing-aware Online Control Framework

In response to the challenges of problem **P1**, we take advantage of *Lyapunov optimization* techniques [12, 13] to design an online control framework.

A. Decomposition using Lyapunov optimization

The key idea of *Lyapunov optimization* techniques is to decompose a long-term stochastic optimization problem into several sub-problems, which can be sequentially solved in each time slot. However, it cannot be directly applied to solve problem **P1**, as the percentile function $P_{q_j}(\cdot)$ imposes tightly coupling among variables of $x_j^E(t)$ or $x_j^I(t)$ over all time slots in a charging period. Thus, we are inspired to first transform problem **P1** to a relaxed optimization problem **P2** that is decomposable. We then decompose the problem **P2** using the *Lyapunov optimization* techniques [12]. Such decomposition follows two steps: *1)* transforming the long-term constraints (Eqs. (5)(6)(7)) into queue stability problems; *2*) constructing the *drift-plus-cost* to divide the relaxed problem **P2** into several sub-problems that can be solved in each time slot. The *drift-plus-cost* can actually characterize the cost-deadline tradeoff.

1) ***A relaxed optimization problem:*** To construct a relaxed problem, we consider that all system statistics including $e_s(t)$, $x_j^E(t)$, and $x_j^I(t)$ cannot be observed till the end of time slot t. Define $\beta_j(t){=}\max\{\beta_j^E(t),\beta_j^I(t),B_j\}$, where $\beta_j^E(t){=}P_{q_j}(x_j(\lfloor t/N \rfloor N),\cdots,x_j(t),0,\cdots,0)$ and similarly $\beta_j^I(t){=}P_{q_j}(x_j(\lfloor t/N \rfloor N),\cdots,x_j(t),0,\cdots,0)$. Then, we have the following problem **P2**:

$$\min_{r_s(t)} \ \widetilde{C}{=}\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\sum_{j=1}^{M}c_j\beta_j(t) \text{ s.t. Eqs. (5)(6)(7)(8).} \quad (9)$$

Although problem **P2** and **P1** have different objectives, **P2** is equivalent to the original problem **P1** in terms of the optimal solution, which is shown in the following theorem.

2) ***Queue-based constraints:*** We construct two groups of actual queues and one group of virtual queues. *Firstly*, to accommodate the constraint in Eq. (5), we construct a group of virtual queues $Q_s(t)$ for each $s{\in}S$. Initially, we define $Q_s(0){=}0, \forall s{\in}S$, and then update the queues in each time slot as follows:

$$Q_s(t+1) = \max\{Q_s(t)+e_s(t)-r_s(t),0\}, \forall s. \quad (10)$$

These virtual queues take $e_s(t)$ as input and $r_s(t)$ as output. They stores the difference in the expected transmission rate and actual transmission rate. The queue lengths are essentially historical deviation from expected rates of the inter-DC flows.

Secondly, we construct a group of actual queues $Q_j^E(t)$ for the uplink of each DC. When inter-DC flows arrive, they are actually stored in queue $Q_j^E(t)$ to await be scheduled. The queueing dynamics are then

$$Q_j^E(t+1) = \max\{Q_j^E(t)-U_j^E+x_j^E(t),0\}, \forall j. \quad (11)$$

For each DC d_j, $Q_j^E(0){=}0$. $x_j^E(t)$ can be viewed as arrivals of queue $Q_j^E(t)$, while the capacity U_j^E can be viewed as the service rate of such a queue. Additionally, for each $d_j{\in}\mathcal{M}$, we call $Q_j^E(t)$ the backlog at time t, as it represents an amount of output bandwidth that needs to be allocated.

Thirdly, we construct another group of actual queues $Q_j^I(t)$ for the downlink of each DC, with the backlog being empty at time slot 0, i.e., $Q_j^I(0){=}0$.

$$Q_j^I(t+1) = \max\{Q_j^I(t)-U_j^I+x_j^I(t),0\}, \forall j. \quad (12)$$

These queues can actually accommodate the constraints in Eqs. (5) (6) (7) if they are stable, as proved in the following.

3) ***Characterizing the cost-deadline tradeoff:*** Let $\boldsymbol{Q}(t)$ denote the concatenated vector of all virtual and actual queues, $\boldsymbol{Q}(t){=}[Q_s(t),Q_j^E(t),Q_j^I(t)]$. Then, we define the Lyapunov function as follows:

$$L(\boldsymbol{Q}(t)){=}\frac{1}{2}\left(\sum_{s=1}^{S}Q_s(t)^2{+}\sum_{j=1}^{M}Q_j^E(t)^2{+}\sum_{j=1}^{M}Q_j^I(t)^2\right). \quad (13)$$

This equality quantitatively reflects the congestion [12] of all queues. On the premise of all queues having strong stability, a small value of $L(\boldsymbol{Q}(t))$ directly implies that queue backlogs are small. To keep the stabilities of queues, the Lyapunov function needs to be persistently pushed towards a lower congestion state. Thus, we are inspired to introduce one-step Lyapunov drift $\Delta(\boldsymbol{Q}(t))$ [12], which is the expected change of queue backlogs over one time slot.

$$\Delta(\boldsymbol{Q}(t)) = \mathbb{E}\left\{L(\boldsymbol{Q}(t+1))-L(\boldsymbol{Q}(t))|\boldsymbol{Q}(t)\right\}. \quad (14)$$

In addition to stabilize the queue backlog to ensure the constraints on deadlines and link capacities, we also need to consider the transmission cost given by the objective function of the problem **P2**. In this case, the problem **P2** can be approximately solved by minimizing the *drift-plus-cost* in each time slot, which jointly considers the queue backlogs and the incurred transmission cost. Mathematically, we have the following sub-problem **P3** in each time slot t

$$\min \ \Delta(\boldsymbol{Q}(t))+V\mathbb{E}\{\sum_{j=1}^{M}c_j\beta_j(t)|\boldsymbol{Q}(t)\}, \text{s.t. Eq. (8).} \quad (15)$$

where V is a control parameter that represents an importance weight on how much we emphasize the transmission cost minimization, compared to constraints on deadline guarantees (Eq. (5)) and link capacities (Eqs. (6) (7)). This provides a flexible design choices among various tradeoff points between transmission cost minimization and deadline guarantees. For example, one may prefer to incur as smaller expected transmission cost as possible, while keeping $\Delta(\boldsymbol{Q}(t))$ small to guarantee deadlines as well as avoid exceeding the bandwidth capacities.

B. Pricing-aware Online Control Algorithm (POCA)

It is easy to check that directly minimizing the objective in Eq. (15) involves unknown backlog information $\boldsymbol{Q}(t{+}1)$. We therefore seek to minimize the its upper bound, without undermining the optimality and performance. Assume that there exist certain peak levels of expected rates e_{max}, such that $e_s(t){\le}e_{max}, \forall s, \forall t$. Then, in each time slot t, given any value of $\boldsymbol{Q}(t)$, the *drift-plus-cost* can be bounded as follows

$$\Delta(\boldsymbol{Q}(t))+V\mathbb{E}\{\sum_{j=1}^{M}c_j\beta_j(t)|\boldsymbol{Q}(t)\}\le H-$$
$$\sum_{j=1}^{M}(Q_j^E(t)U_j^E{+}Q_j^I(t)U_j^I)-\sum_{s=1}^{S}\mathbb{E}\{Q_s(t)(r_s(t){-}e_s(t))|\boldsymbol{Q}(t)\}$$
$$+\sum_{j=1}^{M}\mathbb{E}\{Vc_j\beta_j(t){+}Q_j^E(t)x_j^E(t){+}Q_j^I(t)x_j^I(t)|\boldsymbol{Q}(t)\}. \quad (16)$$

Algorithm 1 Pricing-aware Online Control Algorithm (POCA)

```
1: In the beginning of each time slot $t$, each DC $d_j$ observes the current queue backlogs $Q_j^E(t)$, $Q_j^I(t)$ and $Q_s(t)$ ($\forall s \in S(l_j^I) \vee S(l_j^E)$);
2: Determine the control decisions $r_s(t), \forall s \in S$ to minimize the objective $\sum_{j=1}^{M} \mathbb{E}\{Vc_j\beta_j(t) + Q_j^E(t)x_j^E(t) + Q_j^I(t)x_j^I(t)|\boldsymbol{Q}(t)\} - \sum_{s=1}^{S} \mathbb{E}\{Q_s(t)(r_s(t) - e_s(t))|\boldsymbol{Q}(t)\}$ in problem **P4**.
3: Update the queue backlogs $\boldsymbol{Q}(t)$ according to equalities (10) (11) (12) and the newly determined decisions.
```

where the constant $H \triangleq 2MU_{max}^2 + \frac{1}{2}SU_{max}^2 + \frac{1}{2}Se_{max}^2$, $U_{max} = \max_j\{U_j^I, U_j^E\}$. Due to page limit, the proof process for such upper bound will not be presented here.

With such upper bound, we are essentially solving the following sub-problem **P4** in each time slot t.

$$\min \quad \sum_{j=1}^{M} \mathbb{E}\{Vc_j\beta_j(t) + Q_j^E(t)x_j^E(t) + Q_j^I(t)x_j^I(t)|\boldsymbol{Q}(t)\}$$
$$-\sum_{s=1}^{S} \mathbb{E}\{Q_s(t)(r_s(t) - e_s(t))|\boldsymbol{Q}(t)\} \text{ s.t. Eq. (8)} \tag{17}$$

It should be noted that the sub-problem **P4** is of much smaller scale, and can be efficiently solved by standard optimization solvers. Alternately, it can also be solved in a decentralized fashion by further dividing sub-problem **P4** into multiple per-DC sub-problems that can be independently solved by each DC. In this paper, we do not take in-depth analysis of such decentralized fashion, as it follows a similar idea with the study of [13].

In summary, our **POCA** algorithm is formally summarized in **Algorithm 1**. Specifically, in each time slot t, based on the *online* observation of the queue backlogs $Q_s(t)$, $Q_j^E(t)$, $Q_j^I(t)$, **POCA** strives to solve the problem **P4** for the purpose of determining the transmission rates for all inter-DC flows in time slot t. Finally, **POCA** updates the queue backlogs $Q_s(t)$, $Q_j^E(t)$, $Q_j^I(t)$ according to equalities (10) (11) (12) and the newly determined transmission rates.

V. Performance Evaluation

We evaluate the performance of POCA through small-scale testbed implementation. We build a small testbed with 8 servers to emulate an inter-DC network, with each server representing a DC. All servers are connected to a Pica8 3297 48-port Gigabit switch with an 1Gbps link. Each server has installed Ubuntu, 12.04 64bit version system, and has a 2-core Intel(R) Pentium(R) 3.00GHz CPU, 2GB of RAM, and 1G Ethernet NICs.

With such emulated inter-DC network, we perform distributed per-flow rate limiting on end hosts. At the beginning of each time slot, each end host intercepts all outgoing packets, computes the sending rate for each flow, and marks the sending rate as well as expected rate in the header of each packet. The modified packets are then delivered to Linux Traffic Control (TC) for rate limiting. Like the study of [21], we use two-level Hierarchical Token Bucket in TC to implement rate limit. That is, the root nodes classifies all packets of each flow to a leaf node, and the leaf nodes enforce per-flow rates. From another point of view, the corresponding destination will similarly compute a receiving rate for each flow. Moreover, it will send a feedback to source end host once the receiving rate is smaller than sending rate, such that the source end host can reduce the sending rate to the feedback value.

TABLE I
POCA vs. Default Fair Sharing.

Method	Average	
	Transmission cost ($)	Num. of Failed Flows
POCA	3.91×10^4	4.8
Default Fair Sharing	4.85×10^4	8

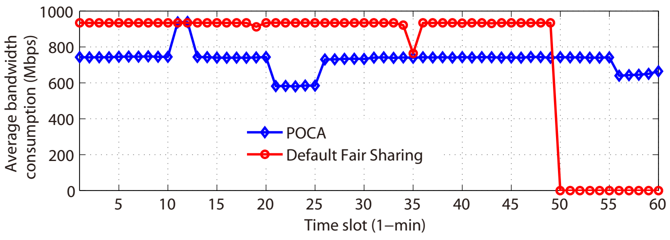


Fig. 3. Average bandwidth consumption.

We conduct our experiments 10 rounds. In each round, it runs one hour, with each time slot being one minute. Following an all-to-all communication pattern, we generate more than 300GB of traffic with 55 flows. Each flow's deadline is set to be a random value within $(0, 60)$ minutes. Similarly, we set the committed bandwidth to be 500Mbps, and use 95-th percentile pricing charging model, for each link. The baseline is the default fair sharing method.

Table I first lists the average transmission cost and the average number of failed flows across all DCs and all experiment times. We can easily check that our POCA can reduce the transmission cost by 19.38% on average, while accommodating more flows with deadlines guaranteed, compared to the default fair sharing. It should be noted that due to the dynamic nature of network, each round of experiment may lead to different number of failed flows. This is why the average number of failed flows may come out to be a decimal. To have a comprehensive understanding of the transmission cost, we plot the average bandwidth consumption across all DCs in Fig. 3. It is clear that POCA achieves a lower bandwidth consumption at most of the time, compared to default fair sharing. Furthermore, POCA can schedule two time slots (e.g., time slot 10 and 11) as traffic peaks, and maintain less traffic differentiation among other time slots. Under the 95-th percentile charging model, the billed bandwidths for POCA and default fair sharing are 747.49Mbps and 935.13Mbps, respectively. These results demonstrate that POCA can practically reduce the transmission cost for inter-DC traffic, while provide acceptable deadline guarantees.

VI. Related Work

Inter-DC traffic engineering has become an active research topic recently. However, none of the existing work can directly solve the problem proposed in this paper.

Regarding the transmission cost on inter-DC traffic, Laoutaris *et al.* [9] present NetStitcher, which uses a store-and-forward approach to schedule inter-DC bulk transfers with the aim of fully utilizing the remaining bandwidth. Similar to NetStitcher, Feng *et al.* present Jetway [10], which conservatively utilize remaining bandwidth for multiple inter-DC video flows. However, they are incapable of accumulating traffic into *"free"* time slots of the q-th percentile charging model. Moreover, neither of the two aforementioned methods considers the traffic deadline. Kandula *et al.* present Tempus [8], which aims to maximize the fraction of transfer delivered before deadline. It achieves fairness among all transfer requests, but does not guarantee the completion of any of them. By taking one step further, Zhang *et al.* propose Amoeba [21], which allows users to explicitly specify the amount of data and deadline. However, neither of them is insufficient to minimize the transmission cost incurred by the inter-DC transfers.

In the context of Internet, there are several studies focused on the impact of percentile charging model. For example, Laoutaris *et al.* propose to use already-paid-for off-peak bandwidth for the delay-tolerant bulk data, and design a source scheduling policy and a store-and-forward policy [7]. Unfortunately, their methods inevitably rely on prior knowledge of the traffic arrivals, and ignore deadlines for the inter-DC transfers. More recently, Golubchik *et al.* study the 95-th percentile minimization problem for scheduling data transfers over the Internet, with constraints on delay requirements [11]. They present both offline and online algorithms to solve this problem. Nevertheless, they assume uniform deadline requirements for all traffic, and care only about the single-sender percentile minimization.

VII. Conclusions

In this paper, we argue that a simple principle of *"more peak, less differentiation"* should be followed when scheduling inter-DC traffic under the *q-th percentile charging model*. To this end, we leverage the diverse deadlines of inter-DC transfer requests and the advantage of Lyapunov optimization technique to design a pricing-aware online control framework. Without a prior knowledge of traffic arrivals, our framework dynamically determines the transmission rates for each inter-DC flow, by efficiently decomposing a long-term optimization into multiple sub-problems that can be sequentially solved in each time slot. To verify the performance of our framework, we conduct small-scale testbed implementation. The results have shown that our framework is capable of reducing the transmission cost, while maintaining satisfactory deadline miss rate. Specifically, it reduces the transmission cost by up to 19.38%, compared to the default fair sharing method.

Acknowledgment

This work is supported by the National Key Research and Development Program of China No. 2016YFB1000205, the State Key Program of National Natural Science of China(Grant No. 61432002), the National Science Foundation for Distinguished Young Scholars of China (Grant No. 61225010), NSFC Grant Nos. 61272417, 61300189, 61370199 and 61672379; Specialized Research Fund for the Doctoral Program of Higher Education (Grant No. 20130041110019), and the Fundamental Research Funds for the Central Universities (Grant. DUT15QY20); the Dalian High-level Talent Innovation Program (No. 2015R049), and the JSPS KAKENHI under Grant 16F16349; the National Natural Science Foundation for Outstanding Excellent young scholars of China under Grant No.61422214, and National Basic Research Program (973 program) under Grant No.2014CB347800.

References

[1] 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 *Proc. of ACM SIGCOMM*, 2013.

[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 *Proc. of ACM SIGCOMM*, 2013.

[3] 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 *Proc. of IEEE INFOCOM*, 2011.

[4] Forrester Research, "The future of data center wide-area networking," http://www.forrester.com.

[5] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 1, pp. 68–73, 2008.

[6] RouteScience Technologies, Inc, "Route optimization for ebusiness applications," White Paper. http://www.routescience.com/, 2003.

[7] N. Laoutaris, G. Smaragdakis, P. Rodriguez, and R. Sundaram, "Delay tolerant bulk data transfers on the internet," in *Proc. of ACM SIGMETRICS*, 2009.

[8] S. Kandula, I. Menache, R. Schwartz, and S. R. Babbula, "Calendaring for wide area networks," in *Proc. of ACM SIGCOMM*, 2015.

[9] N. Laoutaris, M. Sirivianos, X. Yang, and P. Rodriguez, "Inter-datacenter bulk transfers with netstitcher," in *Proc. of ACM SIGCOMM*, 2011.

[10] Y. Feng, B. Li, and B. Li, "Jetway: minimizing costs on inter-datacenter video traffic," in *Proc. of ACM Multimedia*, 2012.

[11] L. Golubchik, S. Khuller, K. Mukherjee, and Y. Yao, "To send or not to send: Reducing the cost of data transmission," in *Proc. of IEEE INFOCOM*, 2013.

[12] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.

[13] F. Liu, Z. Zhou, H. Jin, B. Li, B. Li, and H. Jiang, "On arbitrating the power-performance tradeoff in saas clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 10, pp. 2648–2658, 2014.

[14] Q. Pu, G. Ananthanarayanan, P. Bodik, S. Kandula, A. Akella, P. Bahl, and I. Stoica, "Low latency geo-distributed data analytics," in *Proc. of ACM SIGCOMM*, 2015.

[15] Forrester Research, "Measuring internet congestion: A preliminary report," https://ipp.mit.edu/sites/default/files/documents/Congestion-handout-final.pdf,2014.

[16] B. Briscoe, "Flow rate fairness: Dismantling a religion," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 2, pp. 63–74, 2007.

[17] C.-Y. Hong, M. Caesar, and P. Godfrey, "Finishing flows quickly with preemptive scheduling," in *Proc. of ACM SIGCOMM*, 2012.

[18] M. Alizadeh, S. Yang, M. Sharif, S. Katti, N. McKeown, B. Prabhakar, and S. Shenker, "pfabric: Minimal near-optimal datacenter transport," in *Proc. of ACM SIGCOMM*, 2013.

[19] B. Vamanan, J. Hasan, and T. Vijaykumar, "Deadline-aware datacenter tcp (d2tcp)," in *Proc. of ACM SIGCOMM*, 2012.

[20] D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas, *Dynamic programming and optimal control*. Athena Scientific Belmont, MA, 1995, vol. 1, no. 2.

[21] H. Zhang, K. Chen, W. Bai, D. Han, C. Tian, H. Wang, H. Guan, and M. Zhang, "Guaranteeing deadlines for inter-datacenter transfers," in *Proc. of the Tenth European Conference on Computer Systems*, 2015.