Toward Online Profit-Driven Scheduling of Inter-DC Data-Transfers for Cloud Applications

Ping Lu, Kaiyue Wu, Quanying Sun, Zuqing Zhu[†]
School of Information Science and Technology
University of Science and Technology of China, Hefei, China
[†]Email: {zqzhu}@ieee.org

***Abstract*—For an inter-datacenter (inter-DC) network that carries multiple cloud applications, tasks may arise in the DCs and need to transfer data to others with different latency requirements. Therefore, it is desired that a highly efficient online scheduling algorithm could be developed to consider the request admission, routing selection and bandwidth allocation for the data-transfers jointly. In this work, we investigate this problem, and propose an online scheduling algorithm, namely, *GlobeAny*, to maximize the time-average profit from provisioning the requests. The proposed algorithm leverages Lyapunov optimization techniques and can achieve arbitrarily approaching to the optimal value within $O(1/V)$ gap. We also show that by adjusting the application weights, it can provide differentiated services to requests with different latency requirements.**

***Index Terms*—Inter-DC traffic, Lyapunov optimization, Profit-maximizing, Online scheduling.**

I. Introduction

Nowadays, in order to satisfy the growing demand for various cloud applications, more and more enterprises adopt geographically distributed (geo-distributed) datacenters (DCs) to efficiently deliver high-quality services to the end-users located globally [1–3]. Meanwhile, the large-scale online services, such as e-business, video-on-demand, online gaming, *etc.*, have significantly increased the traffic load in DC networks (DCNs), with both the user-DC and DC-DC communications. It is believed that DC-DC (*i.e.*, inter-DC) communications make the major contribution to the traffic growth in DCNs [2]. Hence, how to efficiently perform traffic scheduling and network capacity allocation for them becomes one of the most challenging problems in geo-distributed DCNs.

Previously, people have studied inter-DC traffic and the corresponding traffic engineering schemes in DCNs [1, 2, 4, 5]. In [1], the authors analyzed the characteristics of inter-DC traffic with commercial datasets. Jain *et al.* proposed to leverage software defined networking (SDN) to manage the traffics among geo-distributed DCs [2]. A network architecture to realize bandwidth-on-demand resource allocation in the optical layer for inter-DC traffic was discussed in [4]. The traffic management schemes for the geo-distributed DCNs that are managed by multiple Internet service providers (ISPs) and the related cost-performance tradeoff were studied in [5]. On the other hand, there are also several studies in the literature [6–9], which are on bulk-data transfer in geo-distributed DCNs. Laoutaris *et al.* proposed a store-and-forward scheme together with time-expanded techniques to minimize the data-transfer time [6]. In [7], the authors adopted max-min fairness in time-expanded networks to ensure service fairness for the data-transfers of multiple cloud applications. By using the minimum-cost multi-commodity flows, Feng *et al.* developed a data-transfer mechanism to reduce the costs for delivering video traffic across DCs [8], and in [9], they extended the mechanism to consider the store-and-forward delivery scheme.

Even though the aforementioned studies on scheduling inter-DC traffic have proposed a few insight ideas and demonstrated interesting results, there are still important issues that have not been given enough attention to. For instance, when trying to reduce the data-transfer cost (determined by the average bandwidth usage, transfer time and price of unit bandwidth) in a DCN, most of the existing approaches usually only considered the cases for certain time periods or several data-sets, but did not target for achieving overall time-average optimality. Basically, for a DCN that carries multiple cloud applications as shown in Fig. 1, tasks may arise in the DCs and need to transfer data to others. Then, in order to achieve efficient online scheduling of the inter-DC traffic generated by these tasks, we need to address the following problems,

- When to provision a task's data-transfer request and where to send the data if multiple DCs can process it?
- Considering the store-and-forward mechanism for data-transfer, how to set up the routes and allocate bandwidth capacities cost-effectively?
- How to balance the tradeoff between request acceptance rate and data-transfer cost and make the scheduling scheme profit-driven?

In this paper, we investigate profit-driven scheduling of inter-DC traffic with a profit model that prices the time-average request acceptance rate and determines data-transfer cost according to the bandwidth usage. We propose an online scheduling algorithm, namely, *GlobeAny*, to schedule the data-transfer requests generated by the inter-DC tasks, which utilizes Lyapunov optimization techniques [10] to maximize the time-average profit. The proposed algorithm achieves arbitrarily approaching to the optimal value within $O(1/V)$ gap, and is evaluated with extensive simulations to show its effectiveness. Moreover, we demonstrate that by adjusting the application weights, the algorithm can provide differentiated services to requests with different priorities and latency requirements.

The rest of the paper is organized as follows. Section II

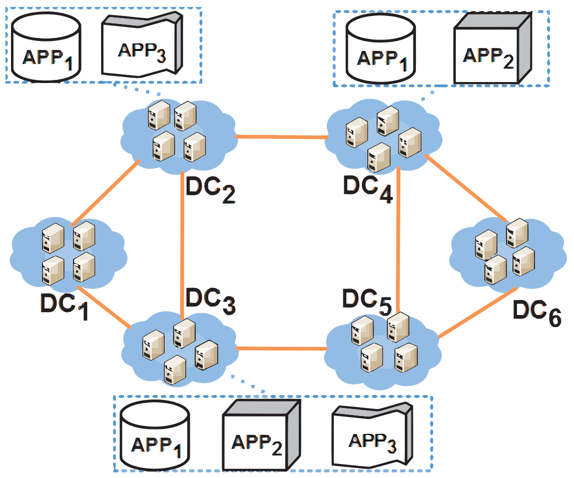

Fig. 1. DCN that carries multiple cloud applications.

describes the system model and formulates the optimization problem for maximizing profit. In Section III, we use Lyapunov optimization techniques to design the online scheduling algorithm, and the performance evaluation is discussed in Section IV. Finally, Section V summarizes the paper.

II. Problem Formulation

A. Network Model

We consider a DCN as the one shown in Fig. 1, and denote it as $\mathcal{G}(\mathcal{D},\mathcal{E})$, which includes $|\mathcal{D}|$ DCs ($\mathcal{D}=\{1,\dots,|\mathcal{D}|\}$ representing the DC set) and the link set $\mathcal{E}$ whose elements inter-connect them. The available bandwidth on link $(i,j)\in\mathcal{E}$, which is between two adjacent DCs i and j, is $b_{i,j}$. We assume that the DCN is a discrete-time system that operates on discrete time-slots (TS') $\{\Delta t,\ 2\Delta t,\ \dots\}$. Here, the time interval Δt can range from a few seconds to several minutes [11], and to adapt to the network dynamics, it should be sufficiently long for the operator to update the traffic scheduling scheme. Note that in the following analysis, we normalize the TS' with Δt and get TS' as $t\in\{1,\ 2,\ \dots\}$.

As Fig. 1 shows, there are numbers of cloud applications (APPs) running in the DCN. We define $\mathcal{K}=\{1,\dots,|\mathcal{K}|\}$ as the set of active APPs, and assume that at TS t, the k-th APP that runs in DC i generates $A_i^k(t)$ inter-DC data-transfer requests. For each request from the k-th APP, the amount of data to be transferred to the destination DC for processing is s_k, and we define A_k^{max} as the maximum value of $A_i^k(t)$,

$$A_k^{max}=\max(A_i^k(t)),\quad \forall i\in\mathcal{D},\ t\in\mathbb{Z}^+.$$

In each DC, we have an arbiter for each APP, and based on the network status, it determines whether a data-transfer request from the APP should be accepted or not. At TS t, the number of accepted requests from the k-th APP in DC i is denoted as $a_i^k(t)$. Then, the accepted requests from the k-th APP are buffered in a queue $Q_i^k(t)$, which is initialized as $Q_i^k(0)=0$. With all the queues for all the APPs across all the DCs, we have a matrix of queues for the DCN at TS t

$$\mathbf{Q}(t)=\begin{bmatrix} Q_1^1(t) & Q_1^2(t) & \cdots & Q_1^{|\mathcal{K}|}(t)\\ Q_2^1(t) & Q_2^2(t) & \cdots & Q_2^{|\mathcal{K}|}(t)\\ \vdots & \vdots & \ddots & \vdots\\ Q_{|\mathcal{D}|}^1(t) & Q_{|\mathcal{D}|}^2(t) & \cdots & Q_{|\mathcal{D}|}^{|\mathcal{K}|}(t)\end{bmatrix}. \tag{1}$$

Apparently, the accepted requests will be no more than the generated ones, as

$$0\le a_i^k(t)\le A_i^k(t),\quad \forall i\in\mathcal{D},\ \forall k\in\mathcal{K}. \tag{2}$$

The time-average expectation of $a_i^k(t)$ can be written as

$$a_i^k=\lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{a_i^k(\tau)\},\quad \forall i\in\mathcal{D},\ \forall k\in\mathcal{K}.$$

B. Data-Transfer with Store-and-Forward Scheme

Note that in a DCN, data-transfer can be realized with the store-and-forward scheme [6], *i.e.*, by leveraging the storage space in intermediate nodes along the data-transfer path to relay the data hop-by-hop. It is known that by utilizing the bandwidth and storage resources jointly in the DCN, this scheme can achieve better data-transfer performance than the one that delivers data end-to-end directly. In this work, we consider a stochastic scheduling strategy that leverages the store-and-forward scheme to provision inter-DC data-transfer requests. Here, we define a decision variable $c_{i,j}^k(t)$ to represent the number of k-th APP's requests whose data is sent from DC i to DC j (i and j are adjacent in the DCN) within TS t. Hence, the corresponding bandwidth utilization should not exceed the available bandwidth on link (i,j),

$$\sum_k c_{i,j}^k(t)\cdot s_k\le b_{i,j},\quad \forall i,j\in\mathcal{D},\ \forall(i,j)\in\mathcal{E},\ \forall k\in\mathcal{K}. \tag{3}$$

And the time-average expectation of $c_{i,j}^k(t)$ is

$$c_{i,j}^k=\lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{c_{i,j}^k(\tau)\},\ \forall i,j\in\mathcal{D},\ \forall(i,j)\in\mathcal{E},\ \forall k\in\mathcal{K}.$$

Then, the total bandwidth usage on link (i,j) on average is

$$C_{i,j}=\sum_k c_{i,j}^k\cdot s_k,\quad \forall i,j\in\mathcal{D},\ \forall(i,j)\in\mathcal{E}.$$

As not all of the DCs can act as the destination DCs to process the data generated by an APP, we define $\mathcal{D}_k$ as the set of feasible destination DCs for the k-th APP. Then, in the queue $Q_i^k(t)$ on DC i where $i\notin\mathcal{D}_k$, the in-queue requests are the newly-accepted ones from local and the just-arrived ones from remote DCs, while the out-of-queue requests are those sent to remote DCs. Fig. 2 illustrates the operations on the queues in a DC. Since a DC's storage space is usually big enough, we assume that all the queues have infinite length. Hence, the queue for the k-th APP on DC i updates from

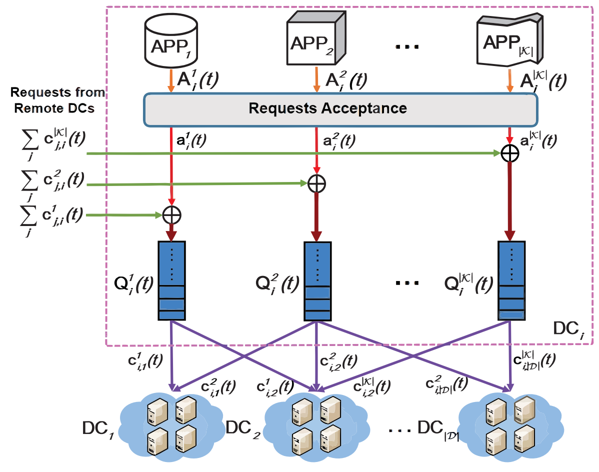

Fig. 2. Queueing operations for data-transfer in DC i.

$Q_i^k(t)$ to $Q_i^k(t+1)$ as

$$Q_i^k(t+1) = \max\left(Q_i^k(t) - \sum_j c_{i,j}^k(t), 0\right) + \sum_j c_{j,i}^k(t) + a_i^k(t), \quad \forall i \in \mathcal{D}/\mathcal{D}_k, (i,j) \in \mathcal{E}, k \in \mathcal{K} \tag{4}$$

Note that compared to the data-transfer process, the data processing in the destination DCs are usually relatively quick since the computing and storage resources in DCs are abundant. Therefore, we assume that when the data of a request arrives at the destination DCs, it is processed immediately. Or in other words, for the queue $Q_i^k(t)$ where $i \in \mathcal{D}_k$, there is no in-queue requests, and it is kept empty all the time, as

$$Q_i^k(t) = 0, \quad \forall i \in \mathcal{D}_k, \ \forall k \in \mathcal{K}. \tag{5}$$

C. Revenue and Cost Models

In order to analyze the tradeoff between the time-average request-acceptance rate a_i^k and the time-average request-transfer rate $c_{i,j}^k$, we formulate the revenue and cost models to calculate the revenue from serving the requests and the cost due to the bandwidth usage. For the revenue model, we consider a logarithmic utility function, similar to [11],

$$f(a_i^k) = log(1 + \alpha_k \cdot a_i^k),$$

where α_k is the fixed revenue coefficient for the k-th APP. The model follows the law of diminishing marginal utility to price the time-average request-acceptance rate for the k-th APP in DC i. Meanwhile, the cost of bandwidth usage on link (i,j) is calculated as

$$g(C_{i,j}) = \beta_{i,j} \cdot C_{i,j},$$

where $\beta_{i,j}$ is the fixed cost of unit bandwidth on link (i,j).

Then, the time-average profit from serving the data-transfer requests is the total revenue minus the total cost, as

$$P = \sum_{i,k} f(a_i^k) - \sum_{i,j} g(C_{i,j}). \tag{6}$$

If we would like to maximize the profit, the optimization problem can be formulated as

$$\begin{aligned} &Maximize \quad P, \\ &s.t. \quad Eqs.\ (2)-(3). \end{aligned} \tag{7}$$

Meanwhile, we should ensure that the lengths of all the queues will not increase towards infinite, *i.e.*,

$$Q_i^k(t) \text{ keeps steady}, \quad \forall i \in \mathcal{D}, \ \forall k \in \mathcal{K}. \tag{8}$$

Note that as the data size per request (*i.e.*, s_k) is bounded, the buffered data in $Q_i^k(t)$ will finite when $Q_i^k(t)$ keeps steady.

III. Online Profit-Driven Traffic Scheduling Algorithm

A. Lyapunov Optimization

In order to maximize the profit from serving the data-transfer requests that generate inter-DC traffic, we design an online profit-driven traffic scheduling algorithm with the aid of Lyapunov optimization [10]. First of all, since the function $f(\cdot)$ is nonlinear, we introduce an auxiliary variable $\gamma_i^k(t)$ as

$$0 \le \gamma_i^k(t) \le A_k^{max}, \quad \forall i \in \mathcal{D}, \ \forall k \in \mathcal{K}. \tag{9}$$

Then, the optimization problem described by Eqs. (7)-(8) can be transformed to a standard stochastic optimization as

$$\begin{aligned} Maximize \quad & \sum_{i,k} \overline{f(\gamma_i^k(t))} - \sum_{i,j} g(C_{i,j}), \\ s.t. \quad & \gamma_i^k \le a_i^k, \quad \forall i \in \mathcal{D}, \ \forall k \in \mathcal{K}, \\ & Eqs. \quad (2), (3), (8), (9), \end{aligned} \tag{10}$$

where

$$\gamma_i^k = \lim_{t \to \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\{\gamma_i^k(\tau)\}, \quad \forall i \in \mathcal{D}, \ \forall k \in \mathcal{K},$$

$$\overline{f(\gamma_i^k(t))} = \lim_{t \to \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\{log(1 + \alpha_k \gamma_i^k(\tau))\}, \ \forall i, \ k.$$

To satisfy the constraint

$$\gamma_i^k \le a_i^k, \quad \forall i \in \mathcal{D}, \ \forall k \in \mathcal{K}, \tag{11}$$

in Eq. (10), we introduce a virtual queue $G_i^k(t)$, which is initialized as $G_i^k(0) = 0$, and use it to transform the constraint to a queue-steady problem according to [10]. The virtual queue $G_i^k(t)$ updates to $G_i^k(t+1)$ as

$$G_i^k(t+1) = \max(G_i^k(t) - a_i^k(t) + \gamma_i^k(t), 0) \ \forall i, \ k. \tag{12}$$

Similar to the definition in Eq. (1), we define a queue matrix for $\{G_i^k(t), \forall i, k\}$ as $\boldsymbol{G}(t)$.

Let $\boldsymbol{\Theta}(t) = [\boldsymbol{Q}(t), \boldsymbol{G}(t)]$, and we define the Lyapunov function $L(\boldsymbol{\Theta}(t))$ as

$$L(\boldsymbol{\Theta}(t)) = \frac{1}{2}\left\{\sum_{i,k}[Q_i^k(t) \cdot w_k]^2 + \sum_{i,k}[G_i^k(t) \cdot w_k]^2\right\}, \tag{13}$$

where w_k is the application weight of the k-th APP. The Lyapunov drift function is the conditional expectation of the Lyapunov function in Eq. (13) for different TS',

$$\Delta(\mathbf{\Theta}(t)) = \mathbb{E}\{L(\mathbf{\Theta}(t+1)) - L(\mathbf{\Theta}(t)) \mid \mathbf{\Theta}(t)\}. \tag{14}$$

By combining the Lyapunov functions and Lyapunov drift functions, we obtain the drift-plus-penalty expression as

$$\Delta(\mathbf{\Theta}(t)) - V\cdot\mathbb{E}\{\sum_{i,k} f(\gamma_i^k(t)) - \sum_{i,j} g(\sum_k c_{i,j}^k(t)\cdot s_k) \mid \mathbf{\Theta}(t)\}, \tag{15}$$

where V is the parameter to balance the queue steadiness and the profit. If we define

$$\eta(t) = \sum_{i,k} f(\gamma_i^k(t)) - \sum_{i,j} g(\sum_k c_{i,j}^k(t)\cdot s_k),$$

and consider that $[\max(q-c,0)+a]^2 \le q^2+a^2+c^2+2q(a-c)$, the drift-plus-penalty expression in Eq. (15) should satisfy the inequation below

$$\begin{aligned}
&\Delta(\mathbf{\Theta}(t)) - V\cdot\mathbb{E}\{\eta(t) \mid \mathbf{\Theta}(t)\} \le B \\
&+ \sum_{i,k}\mathbb{E}\{[Q_i^k(t) - G_i^k(t)]w_k^2\cdot a_i^k(t) \mid \mathbf{\Theta}(t)\} \\
&+ \sum_{i,k}\mathbb{E}\{G_i^k(t)\cdot w_k^2\cdot\gamma_i^k(t) - V\cdot f(\gamma_i^k(t)) \mid \mathbf{\Theta}(t)\} \\
&+ \sum_{i,j,k}\mathbb{E}\{Q_i^k(t)\cdot w_k^2\cdot[c_{j,i}^k(t) - c_{i,j}^k(t)] \\
&\qquad + V\cdot\beta_{i,j}\cdot c_{i,j}^k(t)\cdot s_k \mid \mathbf{\Theta}(t)\},
\end{aligned} \tag{16}$$

where B is a constant that satisfies

$$B \ge \frac{1}{2}\sum_{i,k}\mathbb{E}\left\{\left\{\left[\sum_j c_{i,j}^k(t)\right]^2 + [a_i^k(t) - \gamma_i^k(t)]^2 + \left(\sum_j c_{j,i}^k(t) + a_i^k(t)\right)^2\right\}\cdot w_k^2 \mid \mathbf{\Theta}(t)\right\}.$$

B. Online Traffic Scheduling Algorithm (GlobeAny)

Basically, in order to maximize the profit while keeping the queues steady, we should minimize the right side of the inequation in Eq. (16) according to [10]. Since B is a constant, we need to minimize the rest three terms, *i.e.*,

$$\sum_{i,k}\mathbb{E}\{[Q_i^k(t) - G_i^k(t)]w_k^2\cdot a_i^k(t) \mid \mathbf{\Theta}(t)\}, \tag{17}$$

$$\sum_{i,k}\mathbb{E}\{G_i^k(t)\cdot w_k^2\cdot\gamma_i^k(t) - V\cdot f(\gamma_i^k(t)) \mid \mathbf{\Theta}(t)\}, \tag{18}$$

$$\sum_{i,j,k}\mathbb{E}\{Q_i^k(t)\cdot w_k^2\cdot[c_{j,i}^k(t) - c_{i,j}^k(t)] + V\cdot\beta_{i,j}\cdot c_{i,j}^k(t)\cdot s_k \mid \mathbf{\Theta}(t)\}. \tag{19}$$

Therefore, we design an online scheduling algorithm to minimize the terms in Eqs. (17)-(19). Specifically, with the proposed algorithm, the data for an inter-DC data-transfer request can be forwarded and stored in an intermediate DC until it reaches any of its feasible destination DCs, *e.g.*, for the request from the k-th APP, the feasible destination DCs are those with $i \in \mathcal{D}_k$. Hence, we call the online scheduling algorithm as "*GlobeAny*", and its details are discussed below.

1) Auxiliary Variables: In each TS, the algorithm first determines the auxiliary variables by minimizing the term in Eq. (18). Since the variables $\gamma_i^k(t)$ are independent of each other, we can transform the optimization in this step as

$$\begin{aligned}
&Minimize \quad G_i^k(t)\cdot w_k^2\cdot\gamma_i^k(t) - V\cdot log(1+\alpha_k\cdot\gamma_i^k(t)), \\
&\qquad s.t. \quad 0 \le \gamma_i^k(t) \le A_k^{max}, \quad \forall i \in \mathcal{D},\ \forall k \in \mathcal{K}.
\end{aligned}$$

Then, the optimal auxiliary variables can be obtained by considering the value of $G_i^k(t)$.

- $G_i^k(t) > \frac{V\cdot\alpha_k}{w_k^2}$:
$$(\gamma_i^k(t))^* = 0, \tag{20}$$
- $G_i^k(t) \in \left[\frac{V\cdot\alpha_k}{w_k^2(1+\alpha_k\cdot A_k^{max})},\ \frac{V\cdot\alpha_k}{w_k^2}\right]$:
$$(\gamma_i^k(t))^* = \frac{V}{G_i^k(t)\cdot w_k^2} - \frac{1}{\alpha_k}, \tag{21}$$
- $G_i^k(t) \in \left[0,\ \frac{V\cdot\alpha_k}{w_k^2(1+\alpha_k\cdot A_k^{max})}\right)$:
$$(\gamma_i^k(t))^* = A_k^{max}. \tag{22}$$

2) Requests Acceptance: Similarly, we can get the optimal value for $a_i^k(t)$ (*i.e.*, the number of requests to be accepted in TS t), by minimizing the term in Eq. (17). Hence, the optimization problem is transformed to

$$\begin{aligned}
&Minimize \quad [Q_i^k(t) - G_i^k(t)]\cdot a_i^k(t), \\
&\qquad s.t. \quad 0 \le a_i^k(t) \le A_i^k(t), \quad \forall i \in \mathcal{D},\ \forall k \in \mathcal{K}.
\end{aligned} \tag{23}$$

Then, we can get the optimal value of $(a_i^k(t))^*$ as

$$(a_i^k(t))^* = \begin{cases} A_i^k(t), & Q_i^k(t) \le G_i^k(t), \\ 0, & else, \end{cases} \tag{24}$$

Eq. (24) means that all the generated requests should be accepted when the length of queue $Q_i^k(t)$ is not longer than that of $G_i^k(t)$, otherwise, the generated requests should be dropped to avoid making the queue unsteady.

3) Bandwidth Allocation: In order to minimize the term in Eq. (19), we formulate an optimization problem as

$$\begin{aligned}
&Maximize \quad \sum_{i,j,k}\{c_{i,j}^k(t)\cdot[Q_i^k(t)\cdot w_k^2 - Q_j^k(t)\cdot w_k^2 \\
&\qquad\qquad - V\cdot\beta_{i,j}\cdot s_k]\}, \\
&\qquad s.t. \quad \sum_k c_{i,j}^k(t)\cdot s_k \le b_{i,j}, \quad \forall i,j \in \mathcal{D}, (i,j) \in \mathcal{E}.
\end{aligned} \tag{25}$$

Since we want to make sure that the bandwidth allocated to a DC gets fully utilized, we add a constraint to the optimization problem in Eq. (25) as

$$\sum_j c_{i,j}^k(t) \le Q_i^k(t), \quad \forall i \in \mathcal{D}, (i,j) \in \mathcal{E}, k \in \mathcal{K}. \tag{26}$$

Specifically, Eq. (26) ensures that the bandwidth allocated to all the output links of DC i for the k-th APP will not be larger than the actual data buffered in queue $Q_i^k(t)$.

Even though the optimization problem defined with Eqs. (25)-(26) can be solved with integer linear programming (ILP) tools, the complexity can be relatively high and make the approach not scalable for large-scale problems, *e.g.*, the number of APPs in the DCN is large.

Inspired by the classical Back-Pressure routing algorithm [12], we design a time-efficient heuristic for bandwidth allocation, which is shown in *Algorithm* 1. Here, we define

$$\zeta_{i,j}^k(t) = Q_i^k(t) \cdot w_k^2 - Q_j^k(t) \cdot w_k^2 - V \cdot \beta_{i,j} \cdot s_k, \quad (27)$$

and then, we can get the bandwidth allocation such that all the bandwidth on link (i, j) is allocated to the APP whose $\zeta_{i,j}^k(t)$ is the largest. In *Algorithm* 1, *Line* 1 defines and initializes a flag-matrix to index all the queues in the queue-matrix $\mathbf{Q}(t)$ defined in Eq. (1). In *Lines* 2-5, for each adjacent DC j, we find the APP that has the maximum $\zeta_{i,j}^k(t)$ for the output link (i, j), record the corresponding indices of DCs and APPs, *i.e.*, j and k, and mark the flag $\mathbf{F}(j, k)$ as 1. Finally, *Lines* 6-13 allocate bandwidth in a greedy manner based on $\mathbf{F}$, until all the data in $Q_i^k(t)$ is sent or a link's bandwidth is used up. As the algorithm can be implemented in the distributed manner, the computing complexity is $O(|\mathcal{K}| + |\mathcal{D}|)$.

Algorithm 1 Bandwidth allocation to obtain $c_{i,j}^k(t)$ for DC i

```
Input:
    $\{\zeta_{i,j}^k(t)\},\ \mathbf{Q}(t),\ \{b_{i,j}\},\ \{s_k\}$.
1:  $\mathbf{F} = [\mathbf{0}]_{|\mathcal{D}|\times|\mathcal{K}|}$;
2:  for all $j \in \mathcal{D}$ such that $(i,j) \in \mathcal{E}$ do
3:      $k = \underset{k\in\mathcal{K}}{\text{argmax}}\{\zeta_{i,j}^k(t) \mid \zeta_{i,j}^k(t) > 0\}$;
4:      $\mathbf{F}(j,k) = 1$;
5:  end for
6:  for all $k \in \mathcal{K}$ do
7:      for all $j \in \mathcal{D}$ do
8:          if $\mathbf{F}(j,k) = 1$ then
9:              $c_{i,j}^k(t) = \min(Q_i^k(t), \lfloor \frac{b_{i,j}}{s_k} \rfloor)$;
10:             $Q_i^k(t) = Q_i^k(t) - c_{i,j}^k(t)$;
11:         end if
12:     end for
13: end for
```

4) Queue Update: When the bandwidth allocation is done, we update $Q_i^k(t)$ and $G_i^k(t)$ based on their previous queue-lengths, the auxiliary variable $\gamma_i^k(t)$, the number of accepted requests $a_i^k(t)$, and the bandwidth allocation $c_{i,j}^k(t)$, with Eqs. (4) and (12). This is done for all the queues in $\mathbf{Q}(t)$ and $\mathbf{G}(t)$.

According to the theoretical analysis in [10], we can easily prove that the aforementioned algorithm (*i.e., GlobeAny*) based on Lyapunov optimization can achieve the time-average profit that approaches arbitrarily to the optimal value within $O(1/V)$ gap. Due to the space limitation, we omit the optimality analysis of the proposed algorithm here.

IV. Performance Evaluation

We perform numerical simulations with the 7-node Amazon EC2 Cloud inter-DC topology [8] to evaluate the performance of the proposed algorithm, *i.e.*, *GlobeAny*. In the topology, we set the link capacities with the data set in [8] and scale down the link costs with 10^{-5} proportionally for normalization. In the DCN, the number of APPs is assumed to be $|\mathcal{K}| = 20$, and the amount of data to be transferred for the k-th APP's request is set as $s_k = 2^k$ k-bits. We set the maximum request arrival-rate per TS, A_k^{max}, as inversely proportional to s_k, according to [2], and hence $A_k^{max} = 2^{20-k}$. The set of feasible destination DCs $\mathcal{D}_k$ for the k-th APP is chosen arbitrarily and $|\mathcal{D}_k|$ would not exceed 4. The arrivals of the requests can follow an arbitrary distribution as long as the average arrival-rate per TS equals $\frac{1}{2}A_k^{max}$, and here we use the uniform distribution. The length of a TS is assumed to be 1 minute, and all the results are obtained by averaging the outputs from 10 independent simulations that each contains 10000 TS'. The details on the simulation parameters are listed in Table I.

TABLE I
Simulation Parameters

$\lvert\mathcal{K}\rvert$, number of APPs	20
w_k, weight of the k-th APP	0.01×2^k
s_k, data-size for the request of the k-th APP	2^k k-bits
α_k, revenue coefficient for the k-th APP	0.01×2^k
A_k^{max}, the maximum request arrival-rate of the k-th APP	2^{20-k}
Duration of a TS	1 minute
TS' in each Simulation	10000

A. Impact of the Tradeoff Parameter V

We first conduct simulations to compare the performance of two scenarios of *GlobeAny*, *i.e.*, one uses the ILP (*GlobeAny-ILP*) and the other uses the heuristic in *Algorithm* 1 (*GlobeAny-HEU*) to solve the bandwidth allocation in Subsection III-B3. Fig. 3 shows the results on the time-average profit (*i.e.*, P in Eq. (6)) from the two scenarios, and it can be seen that *GlobeAny-HEU* can provide reasonably good results on P, when compared to *GlobeAny-ILP*, especially when $V \geq 2 \times 10^7$, while the computation time of *GlobeAny-HEU* is around $\frac{1}{100}$ of that of *GlobeAny-ILP*, according to the simulations. The results also indicate that P increases fast when V is relatively small, while the slope decreases gradually afterwards and P eventually converges to the maximum value smoothly. This observation confirms that *GlobeAny* can approach arbitrarily to the optimal value within $O(1/V)$ gap.

Then, we define the average queue-length for the k-th APP as $\overline{Q^k} = \overline{(\sum_i Q_i^k(t))}$, where $\overline{(\cdot)}$ denotes the operation for time-averaging over all the TS' in a simulation. Similarly, the average total queue-length for all the APPs can be defined as $\overline{Q^{total}} = \overline{(\sum_{i,k} Q_i^k(t))}$. Fig. 3 also plots the results on $\overline{Q^1}$ and $\overline{Q^{total}}$. The queue-lengths increase with V, which means that according to the Little's Law, the average delay for data-transfer also becomes longer for a larger V. Hence, we can adjust V to balance tradeoff between the profit and delay.

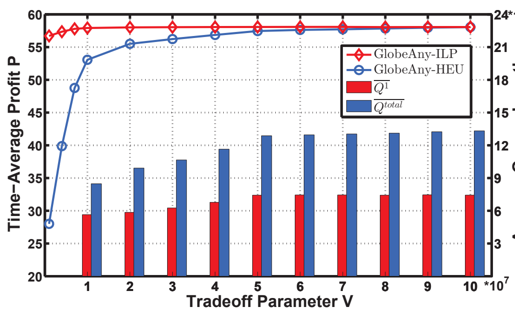

Fig. 3. Impact of the tradeoff parameter V.

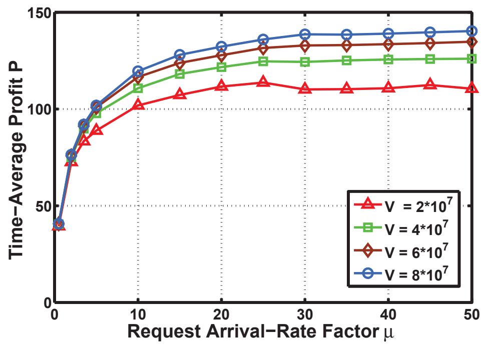

Fig. 4. Impact of the traffic load factor μ.

B. Request Arrival-Rate

In order to evaluate the algorithm's performance for different traffic loads, we multiply the average arrival-rates of each APP's requests by a factor μ, and plot the results on P from *GlobeAny-HEU* in Fig. 4. We observe that P increases with the traffic load first and then converges to a maximum value, which indicates that the algorithm can successfully handle the traffic increase and keep the revenue higher than the cost. The maximum value of P becomes larger for a larger V.

C. Application Weights for Service Differentiation

It is known that the APPs deployed in an inter-DC network may require different levels of quality-of-service (QoS). With *GlobeAny*, we actually can achieve service differentiation by adjusting the application weight of each APP (*i.e.*, $\{w_k\}$). We perform simulations with $|\mathcal{K}| = 40$ APPs to demonstrate this functionality. Basically, we multiply the application weight of the first APP w_1 by a factor θ, while keeping the rest application weights unchanged. Fig. 5 shows the results from *GlobeAny-HEU* algorithm on the time-average queue-lengths for the first and second APPs, *i.e.*, $\overline{Q^1}$ and $\overline{Q^2}$, respectively. It can be seen that with a larger θ, *i.e.*, a larger application weight for higher service priority, $\overline{Q^1}$ decreases dramatically while $\overline{Q^2}$ remains unchanged.

V. Conclusion

This paper studied profit-driven online scheduling of inter-DC data-transfer requests. By leveraging Lyapunov optimization techniques, we proposed an online scheduling algorithm, *i.e.*, *GlobeAny*. It considered the request admission, routing selection and bandwidth allocation for data-transfers jointly, with the objective to maximize the time-average profit. The proposed algorithm could achieve arbitrarily approaching to the optimal value within $O(1/V)$ gap, and we showed that by adjusting the application weights, it could provide differentiated services to requests with different latency requirements.

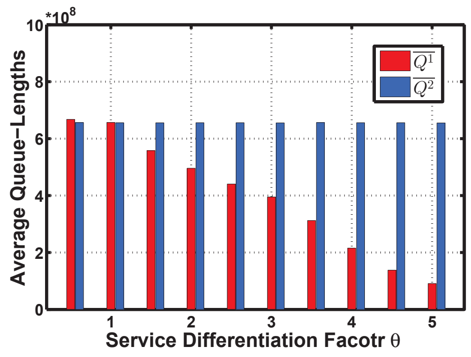

Fig. 5. Service differentiation by adjusting the application weight.

Acknowledgments

This work was supported in part by the Project NCET-11-0884, the NSFC Project 61371117, the Fundamental Research Funds for Central Universities (WK2100060010), and the Strategic Priority Research Program of CAS (XDA06030902).

References

[1] Y. Chen *et al.*, "A first look at inter-data center traffic characteristics via Yahoo! datasets," in *Proc. of INFOCOM 2011*, pp. 1620–1628, Apr. 2011.
[2] S. Jain *et al.*, "B4: Experience with a globally-deployed software defined WAN," in *Proc. of ACM SIGCOMM 2013*, pp. 3–14, Aug. 2013.
[3] X. Yi, F. Liu, J. Liu, and H. Jin, "Building a network highway for big data: architecture and challenges," *IEEE Network*, vol. 28, pp. 5–13, Jul. 2014.
[4] A. Mahimkar *et al.*, "Bandwidth on demand for inter-data center communication," in *Proc. of ACM HotNets 2011*, pp. 24:1–24:6, Nov. 2011.
[5] Z. Zhang *et al.*, "Optimizing cost and performance in online service provider networks," in *Proc. of USENIX NSDI 2010*, pp. 1–15, Mar. 2010.
[6] N. Laoutaris, M. Sirivianos, X. Yang, and P. Rodriguez, "Inter-datacenter bulk transfers with Netstitcher," in *Proc. of ACM SIGCOMM 2011*, pp. 74–85, Aug. 2011.
[7] Y. Wang *et al.*, "Optimal routing and bandwidth allocation for multiple inter-datacenter bulk data transfers," in *Proc. of IEEE ICC 2012*, pp. 5538–5542, Jun. 2012.
[8] Y. Feng, B. Li, and B. Li, "Jetway: Minimizing costs on inter-datacenter video traffic," in *Proc. of ACM MM 2012*, pp. 259–268, Oct. 2012.
[9] ——, "Postcard: Minimizing costs on inter-datacenter traffic with store-and-forward," in *Proc. of ICDCSW 2012*, pp. 43–50, Jun. 2012.
[10] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan and Claypool Publishers, 2010.
[11] F. Liu *et al.*, "On arbitrating the power-performance tradeoff in SaaS clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, pp. 2648–2658, Oct. 2014.
[12] L. Georgiadis, M. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Found. Trends Netw.*, vol. 1, pp. 1–144, Apr. 2006.