IEEE Open Journal of the Communications Society

Received 21 November 2019; revised 19 December 2019; accepted 2 January 2020. Date of publication 7 January 2020; date of current version 24 January 2020.

Digital Object Identifier 10.1109/OJCOMS.2020.2964303

Latency-Constrained Cost-Minimized Request Allocation for Geo-Distributed Cloud Services

XINPING XU [1], WENXIN LI [2], HENG QI [1], JUNXIAO WANG [1], AND KEQIU LI [1] (Senior Member, IEEE)

[1]School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China

[2]Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong

CORRESPONDING AUTHOR: H. QI (e-mail: hengqi@dlut.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB1000205, in part by the State Key Program of National Natural Science of China under Grant 61432002, in part by the National Natural Science Foundation of China–Guangdong Joint Fund under Grant U1701263, and in part by the National Natural Science Foundation of China under Grant 61702365, Grant 61672379, and Grant 61772112.

ABSTRACT Latency to end-users and regulatory requirements push cloud providers to operate many datacenters all around the globe to host their cloud services. An emerging problem under such geo-distributed architecture is to assign each user request to an appropriate datacenter to benefit both cloud providers (e.g., low bandwidth cost) and end-users (e.g., low latency)—known as request allocation. However, prior request allocation solutions have significant limitations: they either focus only on optimizing the benefits for one entity (e.g., providers or users), or ignore some practical yet indispensable factors (e.g., heterogeneous latency requirements of different users and diverse per unit bandwidth cost among different datacenters) when optimizing benefits for both entities. In this paper, we study the problem of minimizing the total bandwidth cost for cloud service providers while guaranteeing the latency requirement for end-users. Specifically, we formulate an integer programming with consideration of the diversities in both the delay of requests and per unit bandwidth cost of datacenters. To efficiently and practically solve this problem, we first relax the integer programming into a continuous convex optimization and then take the advantages of random sampling to enforce the solution to be a feasible one for the original integer programming. We have conducted rigorous theoretical analysis to prove that our algorithm can provide a considerable good competitive ratio. Extensive simulations demonstrate that our proposed algorithm can reduce the total bandwidth cost by 30% while guaranteeing the latency requirements of all requests, as compared to conventional methods.

INDEX TERMS Cloud services, request allocation, latency, bandwidth cost, random sampling and rounding.

I. INTRODUCTION

NOWADAYS, most cloud service providers (e.g., Google, Microsoft, Amazon) have deployed a geographically distributed infrastructure [1], [2], where datacenters are placed at different regions across the world to enhance application robustness and reduce user access delay at the same time. As shown in Figure 1, the user submits service request to the service mapping node. Service mapping nodes can be domain name servers, HTTP proxies, and even software defined controllers. The service mapping node is responsible for directing each request to a specific datacenter to provide cloud services. For example, in Google CDN [3], different data centers have the same copy of contents, and user requests to access that content need to be redirected to one data center.

The problem of distributing user requests among multiple datacenters is known as request allocation, which is critical for both service providers and end-users. On the one hand, if requests are allocated to the datacenter far away from end-users or a near one but with excess workloads, then delay requirements may not be satisfied. On the other hand, non-optimized request allocation strategy will lead to a great waste of bandwidth resources, thereby increasing the operating costs of cloud service providers significantly.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

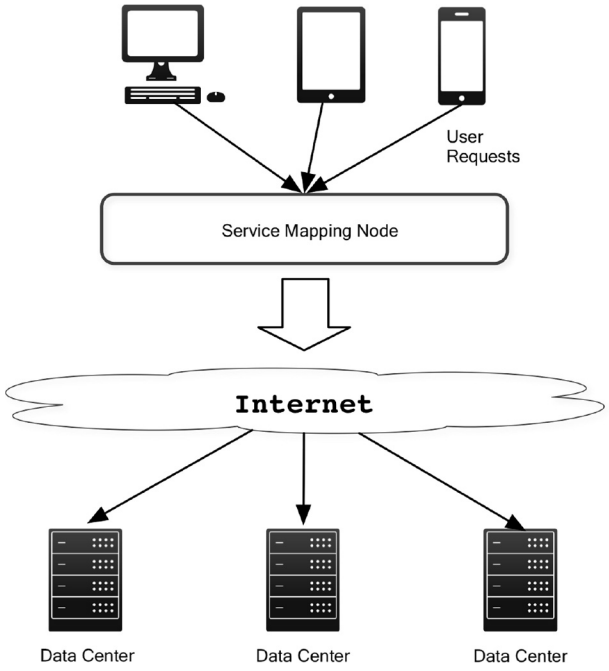


FIGURE 1. User requests allocation diagram across data centers.

Although the request allocation problem is crucial, it also faces the following two challenges. *First*, it is typical that many requests could be submitted simultaneously by different users which may impose different latency requirements [4] to their requests. Moreover, too many requests may make the uplink of a particular datacenter congested, resulting in long queuing delay and even leading to the latency requirements of end users unsatisfied. *Second*, cloud service providers typically make a huge investment in renting bandwidth from ISPs, for traffics from and to their datacenters. Further, as datacenters are located in different regions, a cloud service provider will rent bandwidth from different ISPs with various pricing strategies, forming significant heterogeneity on per unit bandwidth cost across different datacenters [5].

To the best of our knowledge, however, existing work cannot tackle above challenges of the request allocation problem for geo-distributed cloud services. On the one hand, some of existing proposals only focus on minimizing the operation cost for service providers (e.g., [6]–[9]) or simply guaranteeing the quality of service for end users (e.g., [10]–[13]). On the other hand, even though some solutions (e.g., [14], [15]) do optimize the benefits of both service providers and end users, they simply ignore the latency variety from different end users and the diversity on per unit bandwidth cost of different datacenters. Therefore, these programs play a very small role in the actual scene, with regard to minimizing the bandwidth cost for service providers while providing guaranteed performance for end users.

In this paper, we focus on the essential problem of request allocation for geo-distributed cloud services. The primarily goal is to minimize the total bandwidth cost while guaranteeing the latency requirement of each request, when assigning a large amount of concurrent user requests to a set of geographically distributed datacenters owned by any given cloud service provider. Particularly, we first formulate an integer programming. In this problem, we take into account the diverse latency requirement of user requests, the heterogeneity on per unit bandwidth cost of different datacenters, as well as the bandwidth capacity of each datacenter's upstream link. The formulated optimization is a general assignment problem which is typically NP-hard. To solve this problem efficiently, we propose a latency-constrained cost-minimized algorithm. We first relax the integer programming into a convex problem. Based on the optimal solution of this convex problem, we then apply the technique of random sampling to make sure the solution of this convex problem is feasible to the original integer programming problem. By taking a rigorous theoretical analysis, we demonstrate that our algorithm has a good competitive ratio in minimizing the total bandwidth cost. We finally conduct extensive simulations to verify the performance of our proposed algorithm. The results demonstrat the effectiveness of our proposed algorithm on reducing the total bandwidth cost for service providers significantly while guaranteeing the latency requirement of user requests.

The main contributions of this paper are as follows:

- We study the problem of distributing user requests among geo-distributed datacenters to minimize the total bandwidth cost subject to the heterogeneous latency constraints from end-users.
- We develop the mathematical model and formulate an integer programming which considers the diverse per unit bandwidth cost across different datacenters. To solve this problem efficiently, we further develop an algorithm that seamlessly combines the techniques of convex optimization and random sampling, which has a good competitive ratio in minimizing total bandwidth cost.
- We conduct extensive simulations to evaluate the performance of our proposed algorithm, in terms of reducing total bandwidth cost for cloud providers and guaranteeing latency requirements of end-users.

The rest of this paper is organized as follows. In Section II, we develop a mathematical model then the problem formulation. We present the design details of our proposed algorithm in Section III. The simulations results are shown in Section IV. Section V discusses the related work and Section VI concludes this paper.

II. MODELING AND PROBLEM FORMULATION

In this section, we develop a mathematical model to study the problem of minimizing the bandwidth cost for providers while guaranteeing the latency requirement of user requests. Table 1 gives the definition and explanation of the symbolic variables involved in this paper.

TABLE 1. Notations and definitions.

Symbol	Definition
$\mathcal{M}$	The set of datacenters
d_i	datacenter $d_i \in \mathcal{M}$
u_i	The uplink bandwidth capacity of datacenter d_i
c_i	Per unit bandwidth cost of datacenter d_i
$\mathcal{N}$	The set of user requests
r_j	user request $r_j \in \mathcal{N}$
l_j	The latency requirement of user request r_j
b_j	The bandwidth requirement of user request r_j
h_{ji}	The transport delay between d_i and r_j
$P_i(\cdot)$	The response time of datacenter d_i
x_{ji}	Whether datacenter d_i serves request r_j

A. MATHEMATICAL MODEL

Below are the assumptions and simplifications made throughout this paper. Firstly, each of those requests could be directed to an appropriate datacenter, we assume each datacenter hosts the services and contents required by any request which is popular across cloud service providers as discussed in [15]. Such request dispatching can actually be conducted with many tools such as the cloud service gateway, CDN system or centralized software defined controller. In case of CDN, the cache devices that are responsible for providing content services upon user requests would be deployed in the physical edge of the network as the edge layer; when the edge layer fails to hit, it will request the central layer which needs to revert to the source station at the worst case. We also assume that the latency a request experienced contains two parts. The first part is the transport delay between user requests and the service gateway. The second part is the response time within datacenters. Note that our cost model is also simplified, we only care about the bandwidth cost while ignore others which is commonly seen in recent studies such as [5].

We consider a cloud service provider with a set of datacenters, denoted as $\mathcal{M} = \{d_1, d_2, \ldots, d_M\}$. For each datacenter $d_i \in \mathcal{M}$, u_i stands for the bandwidth capacity of its upstream link. In order to indicate the diversity on the bandwidth cost of different datacenters, we use c_i to represent per unit bandwidth cost of datacenter d_i. We consider that there are a set of concurrent user requests during any given time, denoted as $\mathcal{N} = \{r_1, r_2, \ldots, r_N\}$. For each request r_j, b_j is the amount of bandwidth needed with l_j as its latency requirement. We take the transport delay between datacenter d_i and user request r_j as h_{ji}. On the other hand, we denote the response time inside the datacenter as $P_i(\cdot)$ which is a function of the total workload assigned to d_i. While $P_i(\cdot)$ can take various forms depending on the queuing model and the configuration employed on datacenter d_i. It only requires that $P_i(\cdot)$ is an increasing, differentiable and convex function. To indicate decisions on request allocation, we choose x_{ji} as a binary variable representing whether request r_j is assigned to datacenter d_i or not.

B. PROBLEM FORMULATION

We are now in a position to formulate the problem of minimizing the total bandwidth cost for cloud service providers, while guaranteeing the latency requirement for end users when assigning requests to geographically distributed datacenters. This problem is characterized by the following optimization, denoted as **P1**:

$$\min \sum_{d_i \in \mathcal{M}} \sum_{r_j \in \mathcal{N}} b_j c_i x_{ji} \tag{1}$$

$$\text{s.t.} \sum_{r_j \in \mathcal{N}} b_j c_i x_{ji} \le u_i, \ \forall d_i \in \mathcal{M}, \tag{2}$$

$$\sum_{d_i \in \mathcal{M}} P_i\left(\sum_{r_j \in \mathcal{N}} b_j x_{ji}\right) x_{ji} + \sum_{d_i \in \mathcal{M}} h_{ji} x_{ji} \le l_j, \ \forall r_j \in \mathcal{N}, \tag{3}$$

$$\sum_{d_i \in \mathcal{M}} x_{ji} = 1, \ \forall r_j \in \mathcal{N}, \tag{4}$$

$$x_{ji} \in \{0, 1\}, \ \forall d_i \in \mathcal{M}, \forall r_j \in \mathcal{N}. \tag{5}$$

The objective of Eq. (1) for the above optimization problem is to minimize the total bandwidth cost for cloud service provider. Eq. (2) means the total amount of bandwidth consumed on the upstream link of datacenter d_j must not exceed the corresponding bandwidth capacity u_i. Eq. (3) enforces that the latency each request experienced should be bounded by its requirement l_j, where the first term $\sum_{d_i \in \mathcal{M}} P_i(\sum_{r_j \in \mathcal{N}} b_j x_{ji}) x_{ji}$ measures the response time $\sum_{d_i \in \mathcal{M}} h_{ji} x_{ji}$ is the transport delay. Eq. (4) implies each request could only be assigned to one datacenter. Finally, Eq. (5) makes sure that x_{ji} can only take 0 or 1.

Note that **P1** is a comprehensive integer optimization problem whose variables can only take integer values, i.e., 0 or 1. It appears to be in the form of a Generalized Assignment Problem, which is NP-hard or even APX-hard to approximate [16]. Though one may approach the optimal solution by decoupling the problem into several 0-1 knapsack problems, it would introduce high computational complexity [17]. For the sake of practical algorithms design, we are seeking to implement some feasible heuristics alternatively.

III. ALGORITHM DESIGN

In this section, we present the details of our algorithm design. In the following part, we start by transforming the original P1 into a continuous convex optimization problem. Then, we present the technique of random sampling. Finally, we show that our algorithm has a non-trivial competitive ratio for the original problem **P1**.

A. PROBLEM TRANSFORMATION

As mentioned above, **P1** is actually hard to solve. We are therefore motivated to first relax P1 into a continuous convex optimization, which is shown as the following problem **P2**:

$$\min \sum_{d_i \in \mathcal{M}} \sum_{r_j \in \mathcal{N}} b_j c_i x_{ji} \tag{6}$$

$$\text{s.t.} \sum_{r_j \in \mathcal{N}} b_j c_i x_{ji} \le u_i, \ \forall d_i \in \mathcal{M}, \tag{7}$$

Algorithm 1 Latency-Constrained Cost-Minimized Request Allocation

Input:

The amount of bandwidth to handle request r_j: $\{b_j\}$;
The latency requirement of request r_j: $\{l_j\}$;
The transport delay of request r_j to datacenter d_i: $\{h_{ji}\}$;
The uplink bandwidth capacity of datacenter d_i: $\{u_i\}$;
Per unit bandwidth cost of datacenter d_i: $\{c_i\}$;

Output:

Whether datacenter d_i serves request r_j: $\{x_{ji}\}$;

1: Calculate the optimal solution for problem P2 and obtain $\{x_{ji}\}$ where variables may be fractional;
2: **for** each user request $r_j \in \mathcal{N}$ **do**
3: Sample one d_i from the set $\mathcal{M}$ with a probability x_{ji};
4: **if** u_i less than b_j or Eq. (8) of l_j unsatisfied **then**
5: Repeat Step 3;
6: **else**
7: Set $x_{ji} = 1$;
8: Update the bandwidth capacity, i.e., $u_i = u_i - b_j$ and the response time of datacenter d_i, i.e., $P_i(\cdot)$;
9: **endif**
10: **endfor**

$$\sum_{d_i \in \mathcal{M}} P_i \left(\sum_{r_j \in \mathcal{N}} b_j x_{ji} \right) x_{ji} + \sum_{d_i \in \mathcal{M}} h_{ji} x_{ji} \leq l_j, \ \forall r_j \in \mathcal{N}, \tag{8}$$

$$\sum_{d_i \in \mathcal{M}} x_{ji} = 1, \ \forall r_j \in \mathcal{N}, \tag{9}$$

$$0 \leq x_{ji} \leq 1, \ \forall d_i \in \mathcal{M}, \forall r_j \in \mathcal{N}. \tag{10}$$

where x_{ji} is a continuous variable rather than a 0-1 integer. We can easily check the transformed problem **P2** is a comprehensive convex optimization, as its objective and constraint functions are all convex with respect to x_{ji}. Hence, **P2** can be solved efficiently with standard convex programming solvers, such as CVX [18]. However, even with the optimal solution of **P2**, it cannot in turn provide a feasible request allocation strategy for the original problem **P1** directly. To enforce the optimal solution of **P2** is feasible to **P1**, we then propose to leverage the technique of random sampling.

B. RANDOM SAMPLING

The key idea of this technique is to assign user requests to datacenters simply based on a precomputed probability distribution, which actually can be derived from the optimal solution of the convex optimization problem **P2**. Algorithm 1 summarizes the whole procedure.

Algorithm 1 starts from the optimal solution of **P2** (Step 1). Then, in the for loop (Step 2-10), it chooses a datacenter for each request independently. Specifically, it samples one datacenter for each request r_j with a probability x_{ji}. If the sampled datacenter does not have enough bandwidth or it leads the user request to miss its latency requirement, Algorithm 1 will repeat this sampling process till an eligible datacenter found.

Remarks: Our Algorithm 1 has several significant merits. *First*, the dominating overheads of our algorithm are solving the relevant convex optimization **P2** and performing random sampling. The convex problem P2 can be efficiently solved with standard solvers CVX which can return the results within 200 iterations for large-scale problems. Meanwhile, the operation of random sampling only has a time-complexity of $O(N)$. Hence, we conclude that our algorithm requires very little overhead. *Second*, our algorithm can easily be extended to support online using cases. For instance, we can detect a request scheduling event in an online fashion, i.e., if there is new request arriving or existing request being successfully served. Once observing such an event, we can trigger Algorithm 1 to determine request allocation strategy for concurrent user requests. *Finally*, our algorithm can provide a theoretical performance guarantee for minimizing the total bandwidth cost. More specifically, it can provide an upper bound on the total bandwidth cost.

C. ANALYSIS

We now present the theoretical results that our Algorithm 1 can achieve. Specifically, we leverage the following theorems to derive an upper bound for the optimal total bandwidth cost.

Theorem 1 (Lower Bound of the Optimal Total Bandwidth Cost of P1): Define C_{P1} and C_{P2} as the optimal total bandwidth cost for problems **P1** and **P2**, respectively. Then, we have $C_{P2} \leq C_{P1}$.

Proof: We only proves that **P2** is a relaxation of **P1**, as this will directly results in $C_{P2} \leq C_{P1}$. Specifically, for any feasible solution $S \coloneqq \{x_{ji}\}$ of **P1**, we can always find a feasible solution of **P2**, such that the total bandwidth cost in **P2** equals to C_S (the total bandwidth cost related to solution S). To be particular, we can define a solution $S' \coloneqq \{x_{ji'}\}$ of **P2** with following steps:

- First, set $C_{S'} = C_S$;
- Second, for each r_j, set $x_{ji} = 1$, and $x_{ji'} = 0$ for all $i' \neq i$;

Because S is a feasible solution of **P1**, we know that the constructed solution S' can satisfy all constraints in **P2**. This implies that **P2** is a relaxation of **P1**, and thus the theorem is proved. ■

Lemma 1: Assume that $\{a_h\}_{h=1}^{H}$ are independent random variables, such that a_h $(1 \leq h \leq H)$ takes value 0 with probability $1-p_h$ $(0 < p_h < 1)$, and takes v_h with probability p_h. Define $A = \sum_{h=1}^{H} a_h$. Then, for $\hbar \geq 0$, $\lambda \geq 2e$ such that $v_h \leq \hbar$ and $E[A] \leq \hbar$, we have $Pr[A > \lambda\hbar] \leq exp(-\frac{\lambda}{2})$.

Proof: The proof process are similar to that in [19], [20]. We omit the details here. ■

Theorem 2 (Upper Bound of the Total Bandwidth Cost Achieved by Algorithm 1): Suppose P_{avg} is the average response time across all datacenters. Then, the total bandwidth cost achieved by Algorithm 1 is at most

$\max(2\ln 16M, 2\ln 16N)$ times the optimal total bandwidth cost with probability at least 7/8, where M and N are the number of datacenters and requests, respectively.

Proof: For each $d_i \in \mathcal{M}$, we define

$$\rho_i = \frac{\sum_j b_j c_i I(x_{ji} = 1)}{u_i} \tag{11}$$

where $I(\chi)$ is an indicator that equals to 1 if event χ is true and 0 otherwise. Similarly, we define

$$\rho_j = \frac{\left(\sum_i P_i(\sum_{r_j \in \mathcal{N}} b_j x_{ji}) + \sum_i h_{ji}\right) I(x_{ji} = 1)}{l_j} \tag{12}$$

Then, we have $\rho = \max(\max_i \rho_i, \max_j \rho_j)$, which is the smallest scaling factor of x_{ji} to ensure a valid solution to **P2**. In the following, we fix i and j, and focus on deriving the upper bounds of the probabilities that ρ_i and ρ_j exceed $2\ln 16M$ and $4\ln 16N$, respectively.

We first define random variables $X_j = C_i b_j I(x_{ji} = 1)$ and $X = \sum_{j=1}^{N} X_j$, which have the following prosperities:

- Each X_j is independent.
- The random variable X_j takes value of either 0 or $c_i b_j$, and $Pr[X_j = c_i b_j] = x_{ji}$.
- Using constraint (6) in **P2**, we have $E[X] = \sum_{j=1}^{N} C_i b_j I(x_{ji} = 1) \le u_i$.

These prosperities exactly satisfy the needs of Lemma 1. Hence, by setting $\lambda = 2\ln 16M$ and $\hbar = u_i$, we have $Pr[\rho_i > 2\ln 16M] \le \exp(-\ln 16M)$.

Similarly, we define two type of random variables, $Y_i = (P_{avg} + h_{ji}) I(x_{ji} = 1)$ and $Y = \sum_{i=1}^{M} Y_i$. Also, these variables have the following prosperities:

- All Y_i's are independent.
- Y_i is either 0 or $P_{avg} + h_{ji}$, and $Pr[Y_i = P_{avg} + h_{ji}] = x_{ji}$.
- By using constraint (7) of **P2**, we have $E[Y] = \sum_{i=1}^{M} (P_{avg} + h_{ji}) I(x_{ji} = 1) \le l_j$.

Hence, combining Lemma 1, and setting $\lambda = 2\ln 16N$ and $\hbar = l_j$, we yield $Pr[\rho_j > 2\ln 16N] \le \exp(-\ln 16N)$.

Considering the union bound, we have the following inequalities

$$\begin{aligned} Pr\left[\max_i \rho_i > 2\ln 16M\right] &\le \sum_{i=1}^{M} Pr[\rho_i > 2\ln 16M] \\ &\le M\exp(-\ln 16M) \\ &= 1/16 \end{aligned} \tag{13}$$

$$\begin{aligned} Pr\left[\max_j \rho_j > 2\ln 16N\right] &\le \sum_{j=1}^{N} Pr\left[\rho_j > 2\ln 16N\right] \\ &\le N\exp(-\ln 16N) \\ &= 1/16 \end{aligned} \tag{14}$$

With Eq. (13) and (14), we yield the following inequality, by the union bound.

$$\begin{aligned} &Pr[\rho > \max(2\ln 16M, 2\ln 16N)] \\ &\le Pr\left[\max_i \rho_i > \max(2\ln 16M, 2\ln 16N)\right] \\ &\quad + Pr\left[\max_j \rho_j > \max(2\ln 16M, 2\ln 16N)\right] \\ &\le Pr\left[\max_i \rho_i > 2\ln 16M\right] + Pr\left[\max_j \rho_j > 2\ln 16N\right] \\ &= \frac{1}{8} \end{aligned} \tag{15}$$

The theorem can then be inferred. ■

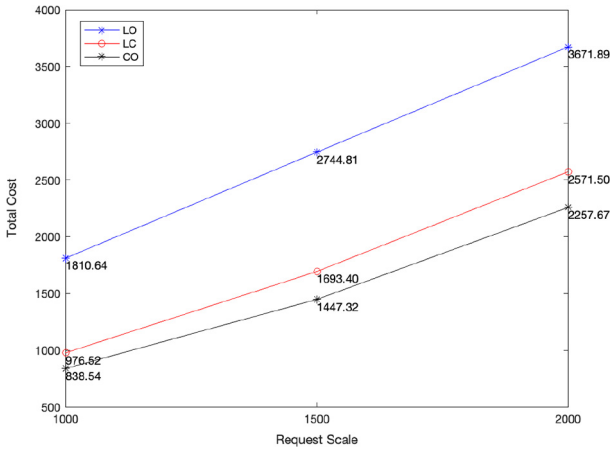


FIGURE 2. Total bandwidth cost generated by different methods.

IV. PERFORMANCE EVALUATION

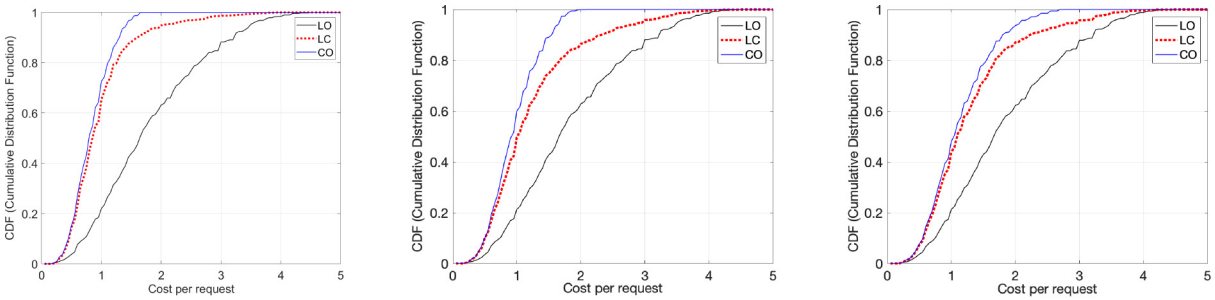
A. SIMULATION SETUP

We simulate a cloud service provider with 40 datacenters. For simplicity, we eliminate the unit of all parameters in our simulation. Particularly, we set the bandwidth capacity of the upstream link for each datacenter as 1000. Per unit bandwidth cost for each datacenter is randomly selected within the range of [0.03, 0.3]. In our simulation, we consider there are 1000, 1500 and 2000 concurrent user requests, respectively and the latency requirement of each request is set between 50 and 500. Also, the amount of bandwidth needed to handle each request is uniformly chosen within the range of [5, 15]. The response time inside each datacenter, i.e., $P_i(\cdot)$ is formulated as its remaining capacity multiplied by a coefficient, which is randomly selected from 1 to 100 initially.

We compare our algorithm with the following two methods. The first one is the latency-only algorithm, which greedily dispatches each request to a datacenter with the lowest overall latency. The second one is the cost-only algorithm which directs each request to a datacenter with the lowest bandwidth cost incurred. To ease the presentation, we denote "LC" as our proposed algorithm, "LO" as the latency-only which ignores costs, whereas "CO" as the cost-only which ignores latency, individually.

B. SIMULATION RESULTS

Total bandwidth cost: Figure 2 shows the total bandwidth cost of different algorithms when 1000, 1500 and 2000 concurrent requests are made separately. It can be observed that



(a) 1000 concurrent requests (b) 1500 concurrent requests (c) 2000 concurrent requests

FIGURE 3. CDF per user request cost.

TABLE 2. The rate of user requests with the latency requirement satisfied.

Algorithm Types	LO	LC	CO
Concurrent 1000	100%	100%	97%
Concurrent 1500	100%	100%	96%
Concurrent 2000	100%	100%	97%

the total cost of LC proposed in this paper is significantly less than that of LO, which is in line with expectations. Because CO only minimizes bandwidth cost for datacenters, while LO only focuses on latency optimization for each request. Specifically, compared with LO, the total cost of LC can be reduced by 30.0% in 2000 concurrent requests, which can also satisfy the latency requirements of all user requests. These results directly show that our proposed algorithm can significantly reduce the total bandwidth cost, when user requests are allocated across geographically distributed data centers.

Latency constraint satisfaction: User requests are also very sensitive to latency experienced. Therefore, this paper also evaluates the overall latency satisfaction of user requests. Comparatively speaking, although LC dose not minimize latency unilaterally, it can fully meet the latency requirements of all requests, as shown in Table 2. Table 2 lists the rate of user requests with the latency requirement satisfied, under different algorithm types. Here we can observe that both LO and LC can guarantee the latency demand of all user requests, while CO cannot. The reason lies in the fact that CO only optimizes the bandwidth cost, without considering the latency requirements of user requests.

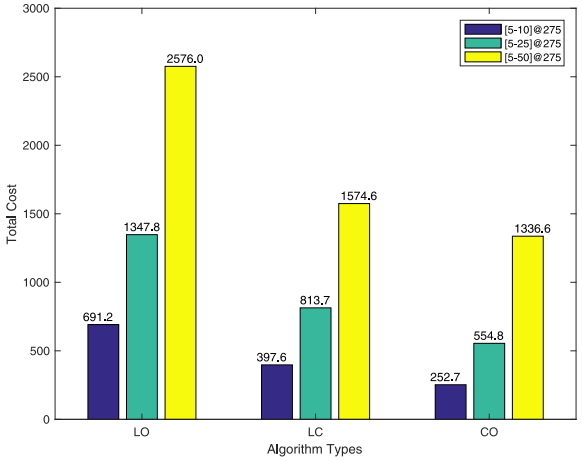
CDF of per user request cost: To further understand costs performance at the micro level, the cumulative distribution functions of the bandwidth cost per user request at 1000 and 2000 concurrencies are plotted in Figures 3. It can be clearly seen that the curve of LC is always on the left side compared with CO and stays very closely to LO for the most of cases, which indicates that our LC algorithm in this paper can always achieve lower bandwidth cost. Taking Figure 3(c) as an example, further observation shows that under this scenario, 80% of user requests of LC corresponds to a lower cost of 1.683; compared with LO algorithm, the cost is reduced by 37.1%; and compared with CO algorithm further, the cost is only increased by 8.5% slightly, considering CO could not 100% satisfy the latency requirement of all user requests.

Impact of bandwidth requirement: So far, the bandwidth requirement of each request is randomly chosen in the range of $[5, 15]$. One may wonder if choosing such bandwidth requirements in a different range can impact the performance of the proposed request allocation algorithm. To answer this point, we conduct multiple experiments. In each experiment, we fix the latency requirement of each request to 275 while varying the range in which the bandwidth requirements are randomly chosen. Fig. 4(a) first shows the total bandwidth cost under different value ranges of the bandwidth requirements. It is clear that under each scenario, our LC algorithm can always achieve relatively low bandwidth costs. Specifically, compared with LO, the total cost of LC can be reduced by as much as 42.5% in 500 concurrent requests when varying the bandwidth requirement b_j from 5 to 10, while being able to satisfy the latency requirements of all user requests. These results directly demonstrate that our algorithm is effective in reducing total bandwidth cost, irrespective of the changing of bandwidth requirement.

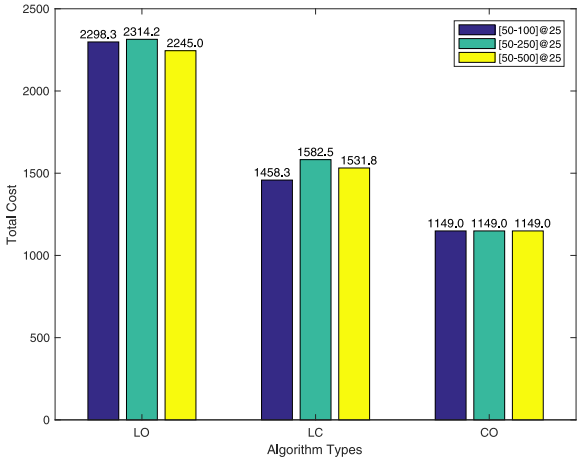
Impact of latency requirement: We next evaluate the impact of latency requirement. To be clear, we randomly choose the latency requirement l_j in the ranges of $[50, 100]$, $[50, 250]$ and $[50, 500]$, respectively, while keeping b_j as a constant value of 25. The results are illustrated in Fig. 4(b), under fixed bandwidth requirement $b_j = 25$. One can easily check it's very similar to the scenarios of before, with the cost reduction around 30% typically comparing to LO. It should be noted that with different ranges of latency requirements, CO would keep at a constant lower value while on the other hand, its delay requirement satisfaction is increasing from 69% to 97% along with the boundary got loosened.

V. RELATED WORK

Request allocation problem for large-scale geo-distributed cloud services has attracted wide attention. This section only reviews the research work closely related to this paper. According to different optimization objectives, existing work can be roughly divided into the following three categories.



(a) Varying bandwidth requirement b_j, with fixed $l_j = 275$



(b) Varying latency requirement l_j, with fixed $b_j = 25$

FIGURE 4. Total cost with 500 concurrent requests when (a) Varying bandwidth requirement b_j, with fixed $l_j = 275$, and (b) Varying latency requirement l_j, with fixed $b_j = 25$.

The first type of research is aimed at optimizing the benefits of cloud service providers, e.g., [6]–[9]. For example, Qureshi *et al.* [6] reduced the energy cost of cloud service providers by designing an efficient service request allocation mechanism. The core basis is to combine the diversity in unit price of different geographical locations. Mathew *et al.* [7] reduced cloud service provider energy consumption and achieve load balancing in the CDN scenario by dynamically powering on/off the CDN server. Gao *et al.* [8] found that the proportion of green energy in different geographic locations can vary significantly, and based on this, an efficient request allocation algorithm was designed to save energy and reduce carbon dioxide emissions. Liu *et al.* [9] achieved green energy-saving and load balancing through service request allocation techniques.

The second type of research mainly focuses on the interests of end users, e.g., [10]–[13]. For example, Wong and Sirer [10] used network detection technology to provide design basis for specific implementation of the nearest service access. Xu and Li [11] proposed DC-FAIR request allocation method based on the Nash equilibrium game theory model, so that the service request of an end user who is far away from the distance can be treated fairly. Zhang *et al.* [12] select service access points nearby for users to reduce the latency of service requests.

However, as the above two kinds of research only honor the benefits of one part, which in turn inevitably affect the interests and performance of the other part.

The last type of request allocation related research considers the interests of both cloud service providers and end users. For example, Xu and Li [14] proposed a distributed ADMM-based request allocation algorithm to jointly balance the performance of cloud providers and their operation cost. Li *et al.* [15] proposed to develop a NBS based model and implement a Logarithmic Smoothing based algorithm to jointly consider high bandwidth utilization for providers and low delay for end users. Liu *et al.* [21] put forward Footprint system for delivering large online services in the "integrated" setting including proxies, data centers and the wide area network. The system optimizes the proxy selection of user requests, the choice of data center as well as the transmission path, which can improve the overall efficiency of the system and reduce the service delay of user requests. Bogdanov *et al.* [22] designed Kurma system for distributed storage services across different regions. The system integrates network latency and response time distribution to accurately estimate the rate of SLO violation when redirecting requests. Kurma can reduce the violation rate, achieve load balancing, and also reduce the service costs. Based on traditional network flow techniques, Fair model is proposed by Xu and Liang [23] to take into account the comprehensive operation cost of the supplier and the SLA of end users, then optimize variable electricity costs.

Although the above methods are all feasible, however, they generally neglect the diversity on the latency requirement from end users, as well as the heterogeneity on per unit bandwidth cost of different data centers. Therefore, while guaranteeing the overall delay, their reduction in the bandwidth cost for cloud service providers is very limited.

VI. CONCLUSION

In this paper, we study an emerging problem of how to allocation each user request to an appropriate data center, aiming at minimizing the total bandwidth cost for cloud service providers while guaranteeing the latency requirements of end users. We formulate an integer programming problem and first relax it into a continuous convex optimization problem, which can be easily solved. Then, we design a request allocation algorithm based on random sampling to ensure that the solution of this optimization problem is feasible to the original one, thus accordingly obtain the decision for request allocation. We have proved that our algorithm can provide a tight upper bound for the total bandwidth cost. Finally,

we conduct comprehensive simulations. The results show that, our proposed algorithm is cost-effective for cloud service providers while guaranteeing the latency requirements of end users, as compared to conventional algorithms.

REFERENCES

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

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

[3] R. Krishnan *et al.*, "Moving beyond end-to-end path information to optimize CDN performance," in *Proc. 9th ACM SIGCOMM Conf. Internet Meas.*, 2009, pp. 190–201.

[4] B. Vamanan, J. Hasan, and T. N. Vijaykumar, "Deadline-aware datacenter TCP (D2TCP)," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 115–126, 2012.

[5] W. Li, K. Li, D. Guo, G. Min, H. Qi, and J. Zhang, "Cost-minimizing bandwidth guarantee for inter-datacenter traffic," *IEEE Trans. Cloud Comput.*, vol. 7, no. 2, pp. 483–494, Apr.–Jun. 2019.

[6] A. Qureshi, R. Weber, H. Balakrishnan, J. V. Guttag, and B. V. Maggs, "Cutting the electric bill for Internet-scale systems," in *Proc. ACM SIGCOMM*, 2009, pp. 123–134.

[7] V. Mathew, R. K. Sitaraman, and P. J. Shenoy, "Energy-aware load balancing in content delivery networks," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, 2012, pp. 954–962.

[8] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, "It's not easy being green," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 211–222, 2012.

[9] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. H. Andrew, "Greening geographical load balancing," in *Proc. ACM SIGMETRICS*, 2011, pp. 233–244.

[10] B. Wong and E. G. Sirer, "ClosestNode.com: An open access, scalable, shared geocast service for distributed systems," *Oper. Syst. Rev.*, vol. 40, no. 1, pp. 62–64, 2006.

[11] H. Xu and B. Li, "A general and practical datacenter selection framework for cloud services," in *Proc. IEEE 5th Int. Conf. Cloud Comput.*, Honolulu, HI, USA, 2012, pp. 9–16.

[12] Z. Zhang, M. Zhang, A. G. Greenberg, Y. C. Hu, R. Mahajan, and B. Christian, "Optimizing cost and performance in online service provider networks," in *Proc. USENIX Conf. Netw. Syst. Design Implement.*, 2010, pp. 33–48.

[13] C. Q. Wu, X. Lin, D. Yu, W. Xu, and L. Li, "End-to-end delay minimization for scientific workflows in clouds under budget constraint," *IEEE Trans. Cloud Comput.*, vol. 3, no. 2, pp. 169–181, Jun. 2015.

[14] H. Xu and B. Li, "Joint request mapping and response routing for geo-distributed cloud services," in *Proc. IEEE INFOCOM*, 2013, pp. 854–862.

[15] W. Li, H. Qi, K. Li, I. Stojmenovic, and J. Lan, "Joint optimization of bandwidth for provider and delay for user in software defined data centers," *IEEE Trans. Cloud Comput.*, vol. 5, no. 2, pp. 331–343, Apr./Jun. 2017.

[16] J. K. Karlof, *Integer Programming: Theory and Practice*. Boca Raton, FL, USA: CRC Press, 2005.

[17] Y. Feng, B. Li, and B. Li, "Bargaining towards maximized resource utilization in video streaming datacenters," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, 2012, pp. 1134–1142.

[18] *CVX Research*. Accessed: Feb. 12, 2019. [Online]. Available: http://cvxr.com/

[19] W. Li, X. Yuan, K. Li, H. Qi, and X. Zhou, "Leveraging endpoint flexibility when scheduling coflows across geo-distributed datacenters," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2018, pp. 873–881.

[20] H. Tan *et al.*, "Joint online coflow routing and scheduling in data center networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 5, pp. 1771–1786, Oct. 2019.

[21] H. H. Liu *et al.*, "Efficiently delivering online services over integrated infrastructure," in *Proc. 13th Usenix Conf. Netw. Syst. Design Implement.*, 2016, pp. 77–90.

[22] K. L. Bogdanov, W. Reda, G. Q. Maguire, Jr., D. Kostić, and M. Canini, "Fast and accurate load balancing for geo-distributed storage systems," in *Proc. ACM Symp. Cloud Comput.*, 2018, pp. 386–400.

[23] Z. Xu and W. Liang, "Operational cost minimization of distributed data centers through the provision of fair request rate allocations while meeting different user SLAs," *Comput. Netw.*, vol. 83, pp. 59–75, Jun. 2015.

XINPING XU is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Dalian University of Technology, China. His research interests include datacenter networks and cloud computing.

WENXIN LI received the B.E. degree from the School of Computer Science and Technology, Dalian University of Technology, China, in 2012, where he is currently pursuing the Ph.D. degree. His research interests include datacenter networks and cloud computing.

HENG QI received the bachelor's degree from Hunan University in 2004 and the master's and Doctoral degrees from Dalian University of Technology, China, in 2006 and 2012, where he was a Lecturer with the School of Computer Science and Technology. He served as a Software Engineer with GlobalLogic-3CIS from 2006 to 2008. He has published more than 20 technical papers in international journals and conferences, including *ACM Transactions on Multimedia Computing, Communications, and Applications* and *Pattern Recognition*. His research interests include computer network, multimedia computing, and mobile cloud computing.

JUNXIAO WANG received the B.S. degree from Dalian Maritime University, Dalian, China, in 2014. He is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Dalian University of Technology, Dalian. His current research interests include software defined network, network function virtualization, and cloud computing.

KEQIU LI (Senior Member, IEEE) received the bachelor's and master's degrees from the Department of Applied Mathematics, Dalian University of Technology in 1994 and 1997, respectively, and the Ph.D. degree from the Graduate School of Information Science, Japan Advanced Institute of Science and Technology in 2005.

He also has two-year postdoctoral experience with the University of Tokyo, Japan. He is currently a Professor with the School of Computer Science and Technology, Dalian University of Technology, China. He has published more than 100 technical papers, in journals such as IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS (TPDS), ACM TOIT, and ACM TOMCCAP. His research interests include Internet technology, data center networks, cloud computing, and wireless networks. He is an Associate Editor of the IEEE TPDS and the IEEE TRANSACTIONS ON COMPUTERS.