Big Data and Information Analytics
©American Institute of Mathematical Sciences
Volume **1**, Number **1**, January **2016**
doi:10.3934/bdia.2016.1.15
pp. **15–29**

TOWARDS BIG DATA PROCESSING IN CLOUDS: AN ONLINE COST-MINIMIZATION APPROACH

Weidong Bao*

College of Information System and Management
National University of Defense Technology
Changsha, Hunan, 410073, China

Wenhua Xiao†, Haoran Ji†, Chao Chen†
Xiaomin Zhu† and Jianhong Wu†,‡

† College of Information System and Management
National University of Defense Technology
Changsha, Hunan, 410073, China
‡ Department of Mathematics and Statistics, York University
Toronto, Ontario, M3J 1P3, Canada

Abstract. Due to its elastic and on-demand nature of resource provisioning, cloud computing provides a cost effective and powerful technology for the processing of big data. Under this paradigm, Data Service Provider (DSP) may rent geographically distributed datacenters to process their large amount of data. As the data are dynamically generated and the resource pricing varies over time, moving the data from differently geographic locations to different datacenters while provisioning adequate computation resource to process them is an essential task to achieve cost effectiveness for DSP. In this paper, a joint online approach is proposed to address this task. We formulate the problem into a joint stochastic optimization problem, which is then decoupled into two independent subproblems via the Lyapunov framework. Our method is able to minimize the long-term time average cost including computing cost, storage cost, bandwidth cost and latency cost. Theoretical analysis shows that our online algorithm can produce a solution within an upper bound to the optimal solution achieved through offline computing and guarantee that the data processing can be completed with preset delays.

1. **Introduction.** The cloud computing paradigm offers a convenient way for users to dynamically adjust its computing resources rented from cloud service providers (CSPs) according to the demand in a Pay-As-You-Go (PAYG) manner. Specifically, in cloud computing, benefited from the development of virtualization technology[3], VMs (Virtual Machines) resources can be scaled up and down to match the applications demands. Compared with traditional approaches, the cloud computing

2010 *Mathematics Subject Classification.* Primary: 90B15, 90B22; Secondary: 68W15, 68W27.
Key words and phrases. Big data, cloud computing, resource scheduling, data allocation, Lyapunov optimization.

This research is supported by the National Natural Science Foundation of China under grant 61405252,61572511, the Hunan Provincial Natural Science Foundation of China under grant 2015JJ3023 as well as the Overseas, Hongkong&Macau Scholars Collaborated Research Fund of China under grant 11428101.

* Corresponding author: Weidong Bao. Email: wdbao@nudt.edu.cn.

paradigm eliminates users' costs of purchasing and maintaining their own infrastructures.

The elastic and on-demand nature of resource provisioning attracts a lot of users to deploy their applications, especially computation-intensive big data analysis in the clouds. At the age of big data, data analysis is more and more important for applications such as financial analysis, social interaction web sites, astronomical telescope service. For example, Facebook-like social media sites can uncover usage patterns and hidden correlations by analyzing the web site history records (e.g., click records, activity records et al.) to facilitate its marketing decision. We call this kind of organization as Data Service Provider (DSP) in this paper. Under this paradigm, the DSPs should solve two problems in the first place: 1) How to transfer the large-scale data sets from various locations into clouds and 2) How many resources such as computing resource and storage resource should be rented in the clouds for processing?

Although much efforts has been made to design computing models for fast big data analysis, such as Mapreduce[6] and Spark[27], the problems of moving large-scale data to the clouds and provisioning adequate resources at the same time in the clouds is rarely considered in the community. Currently, for the data moving problem, practices such as copying the data into large-scale hard drives for physically transportation[2, 15] and even moving the entire machine [1] to datacenters are adopted. These methods not only incur undesirable delays but also insecure case, given that hard drives may be damaged from transportation accident. For the resource provision problem, some works have been done to copy with dynamic workload in clouds[16, 21]. But these methods often considered the data moving problem and resource provisioning problem in isolation.

In this paper, targeting the analysis of big data from different locations with the MapReduce-like framework in the clouds, we propose an online approach which systematically address the data moving problem and resource provisioning problem, with the goal of over all cost minimization of running big data analytic in the couds. To achieve this goal, we first formulate the problem into a jointly stochastic optimization problem, and then, apply the Lyapunov Optimization framework. Such a stochastic system does not require predicting the future system states and makes decisions only based on current system state[13]. Based on the drift-plus-penalty function transformation, we propose an online algorithm that is able to move data from multiple regions to distributed datacenters in an online manner and dynamically rent the near optimal number of computing resource and storage resource needed to satisfy user requirements for serving data analysis.

The major contributions of this work are summarized as follows:

- We propose a novel framework that systematically handles data moving from multiple locations to multiple datacenters and resource renting in each datacenter in a nearly optimal manner. In particular, we consider the bandwidth cost, computing cost, storage cost and delay cost as the overall cost and guarantee the data can be processed within a desirable delay. In our framework, VMs in the cloud have different types and are priced dynamically.
- We propose an algorithm to solve the jointed stochastic problem using the Lyapunov optimization framework, which is able to make decisions of resource renting and data moving online. Moreover, the algorithm can have a distributed implementation.

- We conduct performance analyses for the algorithm theoretically, which demonstrate that the algorithm approximates the optimal solution within provable bounds and is capable of processing the tasks within a preset delay.

The remainder of this paper is organized as follows: Section 2 summarizes related works; Section 3 describes the system modeling and the problem formulation; Section 4 gives the online algorithm for solving the problem; Section 5 analyzes the proposed algorithm; Section 6 concludes the paper.

2. **Related work.** Recent years have witnessed the proliferation of cloud-based service in both academic and industry. Much efforts has been made to migrate the applications such as cloud-based live streaming [9, 21], cloud-based online game [18], cloud-based conference [7] and social media applications[24] etc. into clouds. Majority of these studies have focused on how to scale up and down the resource in the clouds to meet user demand or migrate the workflow into clouds.

Few studies have been conducted to move large scale data into clouds. Paper [4] studied how to transfer data to the cloud provider via the Internet and courier services. Study [5] proposed a solution to minimize the transfer latency under a budget constraint. In [11], the authors studied the data streaming storage for real time big data processing. Different from our study, these work deal with the data transfer problem on static scenario in which the data amount is fixed, while our work consider dynamically generated data. In addition, aforementioned studies considered a single datacenter while our work takes into account multiple datacenters. The most relevant work is Zhang et al [28] which proposed an online algorithm to migrate dynamically generated data from various locations to the clouds for processing. However, our work significantly differs since we consider the resource provisioning and data moving as simultaneously and applied the Lyapunov framework to address the problem.

There is also a line of research on resource provisioning in clouds. In the clouds, the server pool and the capacity of each server become elastic. Studies [16, 12] considered elastic server capacity supported by virtualization technologies. Work [16] proposed adaptive request allocation and service capacities scaling mechanism mainly to cope with the flash crowd. Study [23] took into account of the VM renting cost and storage cost when making scheduling decisions. Different from these works which often need certain mechanisms to predict the future workloads, our work does not rely on any future information on big data tasks since the Lyapunov optimization framework is adopted. Also, studies on how to scheduling the tasks with different objectives in clouds have been conducted. Works [29, 31] proposed efficient scheduling strategies for real-time tasks with energy minimization while studies [31, 20] developed task scheduling algorithms under the consideration of fault-tolerant. These works are often within one single datacenter.

In addition, the Lyapunov optimization technique was first proposed in [17] to address the network stability problem and then was introduced into cloud computing to deal with job admission and resource allocation problem [19, 10]. Yao et al. [26] extends it from the single time scale to two-time-scale for achieving electricity cost reduction in geographically distributed datacenters. Recently, this approach is used for resource management in cloud-based video service [22, 25]. In our work, we utilize this approach to simultaneously address the data moving from multiple locations to multiple datacenters and resource provisioning in each datacenter.

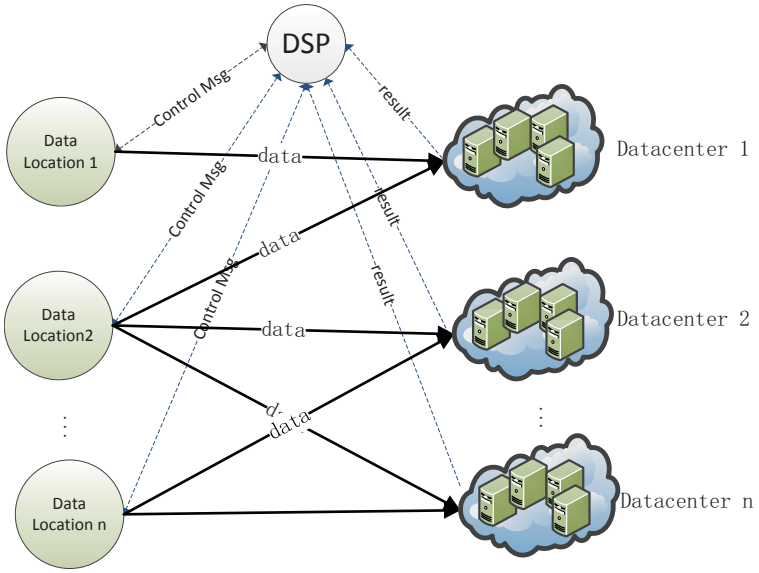

FIGURE 1. System architecture

To summarize, our work differs from existing works as follows. 1) Firstly, we address the problems of data moving and resource provisioning systematically and design an online algorithm that can be implemented distributedly. 2) Secondly, with the Lyapunov framework, our method does not rely on the prediction of future big data processing workload, which significantly differs from the assumptions made in [21, 23].

3. Modeling and formulation.

3.1. System modeling.

We consider such a system scenario as presented in Fig. 1: A DSP (e.g., a global astronomical telescope department) manage multiple geographical data locations that continuously produces large volumes of data. The DSP deploys their data analytics application in cloud and connects the data source to different datacenters located in multiple places. All the data are moved to the the datacenters and processed in the corresponding datacenter with distributed computing model such as MapReduce framework. In the system, the DSP observes the state of the datacenter (e.g., VM price, datacenter load state, network state) and decides the amount of data to be moved to each datacenter and the amount of resource rented from each datacenter, with cost minimization consideration. Finally, the datacenters return the analysis results to DSP after the data have been processed and analyzed.

Formally, considering the geo-distributed datacenters set $\mathcal{D}$ with size of $D = |\mathcal{D}|$, indexed by $d(1 \leq d \leq D)$. A set $\mathcal{K}$ of distinct types of VMs (with size $K = |\mathcal{K}|$), each with specific capacity v_k under different configurations of CPU and memory, are provided in each datacenter. Data are dynamically generated from $R = |\mathcal{R}|$ different data location (indexed by $r, 1 \leq r \leq R$), denoted as set $\mathcal{R}$. Data from any location can be moved to any datacenter for analytics via virtual private networks (VPNs). And the data transmission bandwidth between a data generation location

TABLE 1. IMPORTANT NOTATIONS

$\mathcal{D}$	set of datacenters distributed over multiple regions
$\mathcal{R}$	set of data locations
$\mathcal{K}$	set of VM types
$a_r(t)$	amount of the data generated from region r at t
A^r_{max}	max amount of data generated from region r
$\lambda^d_r(t)$	amount of the data allocated to d from region r at t
$n^{k,\max}_d$	max number of VMs of type-k in datacenter d
$n^k_d(t)$	number of type-k VM provisioned in datacenter d at t
$p^k_d(t)$	price of type-k VM in datacenter d at t
s_d	price of storage in datacenter d
b^d_r	price of bandwidth between location r and datacenter d
v_k	data processing rate of type-k VM
ε_d	preset constant for controlling queueing delay in $H_d(t)$
l	max delay of data process
$H_d(t)$	unprocessed data in datacenter d at t
$Z_d(t)$	Virtual queue associate with $H_d(t)$ to guarantee its delay

$r \in \mathcal{R}$ and a datacenter $d \in \mathcal{D}$ is large as well. To be realistic, we also assume that the bandwidth B_{rd} on a VPN link (r, d) from data location r to data center d is limited, and constitutes the bottleneck in the system. In addition, The data generation in each location is independent and the prices of the resource (e.g., VM, storage) in each datacenter are varying in both spatial and temporal domain.

The system operates according to time slots, denoted by $t = 0, 1, ..., T$. In every time slot, the DSP need to make the decision of moving how much data from data location r to datacenter d and renting how many resources to support its data processing, storage from each datacenter. Our goal is therefore to minimize the over all cost of big data analytics in clouds as well as guarantee the delay in the long run. For ease of reference, important notations are summarized in Table 1.

3.2. **Problem formulation.** In this subsection, we first formulate the cost incurred in the system and then define the objective of the problem mathematically.

As aforementioned, the system runs in a time-slotted fashion and the data are dynamically generated over different regions in each time slot. Let $a_r(t)$ be the amount of data generated from the r-th region at time slot t. Since the data generated from each location can be moved to any datacenter for analytics, we denote $\lambda^d_r(t)$ as amount of the data allocated to d from region r at t and $A^r_{\max}$ as the max number of data generated in location r. Hence, we have:

$$a_r(t) \leq A^r_{\max}, \forall r, t \in [1, T]. \tag{1}$$

$$a_r(t) = \sum_{d \in \mathcal{D}} \lambda^d_r(t), \forall r, t \in [1, T]. \tag{2}$$

The goal of the DSP is to minimize the over all cost incurred in the system by optimizing the amount of data allocated to each datacenter and the number of resources needed. Specifically, the following cost components are considered in this paper: bandwidth cost, latency cost, storage cost and computing cost. Each of the cost is defined as follows.

(1) Usually, the bandwidth price is varying over different VPN links because they often belong to different Internet service providers. Let b^d_r be the price of

transferring 1 GB data between data location $r \in \mathcal{R}$ and datacenter $d \in \mathcal{D}$, then the bandwidth cost of the at t is:

$$C_b(t) = \sum_{d\in\mathcal{D}} \sum_{r\in\mathcal{R}} \lambda_r^d(t) \cdot b_r^d. \tag{3}$$

(2) Storage cost is an important factor to be considered in choosing the datacenter for data analytics since it often has large amount of data for big data application. Let s_d represent the price of storing 1 GB data for one time slot period in datacenter $d \in \mathcal{D}$, then the storage cost at t is:

$$C_s(t) = \sum_{d\in\mathcal{D}} \sum_{r\in\mathcal{R}} \lambda_r^d(t) \cdot s_d. \tag{4}$$

(3) Due to the variance of VM price over time slots, the number of VMs rented from datacenter has important impact on the over all cost of the system as well as QoS of the big data application. Let $n_d^k(t)$ be the number of VMs rented from datacenter d at time slot t. $p_d^k(t)$ be the type-k VM price in datacenter d at time slot t, which is diverse in both spatial and time space. Then the computing cost is defined as follows.

$$C_p(t) = \sum_{d\in\mathcal{D}} \sum_{k\in\mathcal{K}} n_d^k(t) \cdot p_d^k(t). \tag{5}$$

(4) The latency incurred by upload data to the datacenters is an important performance measure, which is to be minimized in the data moving process. L_r^d is the latency between the data location $r \in \mathcal{R}$ and the datacenter $d \in \mathcal{D}$. These delays are determined by the respective geographic distance and can be obtained by a simple command such as Ping. As suggested in [28], we convert the latency into monetary cost. Therefore, we can define the latency cost as:

$$C_l(t) = \sum_{d\in\mathcal{D}} \sum_{r\in\mathcal{R}} \alpha \cdot \lambda_r^d(t) \cdot L_r^d(t), \tag{6}$$

where α is a weight converting latency into a monetary cost.

Based on above cost formulation, the overall cost incurred in the system can be derived as:

$$C(t) = C_p(t) + C_s(t) + C_b(t) + C_l(t). \tag{7}$$

Hence, the problem of minimizing the time-average cost of data moving and processing within a long-term period $[0, T]$ can be formulated as:

$$\mathbf{P1}.\ \min: \quad \overline{C} \tag{8}$$

$$\text{s.t.}: \quad a_r(t) \le A_{\max}^r, \forall r, t \in [1, T] \tag{9}$$

$$a_r(t) = \sum_{d\in\mathcal{D}} \lambda_r^d(t), \forall r, t \in [1, T] \tag{10}$$

$$0 \le n_d^k(t) \le n_d^{k,\max}, \forall d, \forall k, t \in [1, T] \tag{11}$$

where $\overline{C} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T-1} \mathbb{E}\{C(t)\}$. The constraint (10) is to ensure that the sum of data allocated to each datacenter at one time slot is equal to the total amount data generated at that time slot. The constraint (11) ensures that the number of VMs required is within the capacity that a datacenter can provide.

From the problem formulation presented above, as the data generation is unknown, we know that the problem is a constrained stochastic optimization problem and our objective is to minimize the long-term average cost by optimizing the

amount of data allocated to each datacenter as well as the number of the VMs rented inthe datacenter. To deal with this problem, a recent developed optimization technique is adopted in this paper. The details of solution by using Lyapnov optimization framework is presented in the next section.

4. **Online algorithm design.** In this section, we exploit the Lyapunov optimization theory to design our online control algorithm. An outstanding feature of this method is that it does not require future information about workload. By greedily minimizing the drift-plus-penalty in each time slot, it also can be proved to approach a time averaged cost that is arbitrarily close to optimum, while still maintaining system stability.

According to the standard optimization framework theory in [13], we first transform the problem **P1** to an optimization problem of minimizing the Lyapunov drift-plus-penalty term and then design the corresponding online algorithm.

4.1. **Problem transformation.** Let $H_d(t)$ be the amount of unprocessed data in datacenter d at time slot t. Initially, we define $H_d(0) = 0$, and then the evolution of the queue $H_d(t)$ can be described as below:

$$H_d(t+1) = \max[H_d(t) - \sum_{k\in\mathcal{K}} n_d^k(t)\cdot v_k, 0] + \sum_{r\in\mathcal{R}} \lambda_r^d(t). \tag{12}$$

The above queue update implies that the amount of departed data and newly-arrived data are $\sum_{k\in\mathcal{K}} n_d^k(t)\cdot v_k$ and $\sum_{r\in\mathcal{R}} \lambda_r^d(t)$, respectively.

To guarantee that the worst-case queuing delay in queue $H_d(t)$, $\forall d \in \mathcal{D}$, is bounded by the max workload delay l, we design a related virtual queue $Z_d(t)$ according to the ε-persistent service technique for delay bounding in [14]. Similarly, the backlog of virtual queue $Z_d(t)$ is initialized as $Z_d(0) = 0$, then updated as follows:

$$Z_d(t+1) = \max[Z_d(t) + 1_{\{H_d(t)>0\}}(\varepsilon_d - \sum_{k\in\mathcal{K}} n_d^k(t)\cdot v_k) - 1_{\{H_d(t)=0\}} \sum_{k\in\mathcal{K}} n_d^{k,\max}\cdot v_k, 0], \tag{13}$$

where the indicator function $1_{\{H_d(t)>0\}}$ equals to 1 when $H_d(t) > 0$, and 0 otherwise. Similarly, $1_{\{H_d(t)=0\}}$ equals to 1 when $H_d(t) = 0$, and 0 otherwise. ε_d is preset constant that can be gauged to control the queuing delay bound. It can be proved that we are able to guarantee all data can be processed with delays at most l time slots if the algorithm can guarantee the length of $H_d(t)$ and $Z_d(t)$ over time slots. It is also proved that l can be set as $l = [(H_d^{\max} + Z_d^{\max})/\varepsilon_d]$, where $H_d^{\max}$ and $Z_d^{\max}$ are the bound of queues $H_d(t)$ and $Z_d(t)$ respectively. Details please see in theorem 5.3.

Let $\mathbf{Z}(t) = (Z_d(t))$, and $\mathbf{H}(t) = (H_d(t)), \forall d \in \mathcal{D}$ denote the matrix of virtual queue and actual queue respectively. Then, we use $\Theta(t) = [\mathbf{Z}(t), \mathbf{H}(t)]$ to denote the combined matrix of actual queues and virtual queues. According to Lyapunov framework [13], we define the Lyapunov functions as follows:

$$L(\Theta(t)) = \frac{1}{2}\sum_{d\in\mathcal{D}} \{Z_d(t)^2 + H_d(t)^2\}, \tag{14}$$

where $L(\Theta(t))$ measures the queue backlogs in the system. The one-slot Lyapunov drift is:

$$\Delta(\Theta(t)) = \mathbb{E}\{L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)\}. \tag{15}$$

In the sense of Lyapunov optimization framework, the drift-plus-penalty can be obtained by adding the the cost incurred by the system to the above Lyapunov drift, namely,

$$\Delta(\Theta(t)) + V \cdot \mathbb{E}\{C(t)|\Theta(t)\}, \tag{16}$$

where V is a non-negative parameter, that can control the tradeoff between the system stability and cost. The larger the V is, the smaller the cost is, and vice versa. Hence, the original problem **P1** can be transformed into the following problem **P2**:

$$\mathbf{P2}.\ \min: \quad (16) \tag{17}$$

$$\text{s.t.}: \quad (9)(10)(11). \tag{18}$$

To solve problem **P2**, rather than directly minimize the drift-plus-penalty expression (16), we seek to minimize the upper bound for it, without undermining the optimality and performance of the algorithm according to [13]. Therefore, the key is to find an upper bound on problem **P2**. It can be proved that the expression (16) is bounded as:

$$\begin{array}{l} \Delta(\Theta(t)) + V \cdot \mathbb{E}\{C(t)|\Theta(t)\} \\ = B + \mathbb{E}\{\sum\limits_{d\in\mathcal{D}}\sum\limits_{k\in\mathcal{K}} n_d^k(t)\cdot(Vp_d^k(t) - H_d(t)v_k - Z_d(t)v_k)|\Theta(t)\} \\ +\mathbb{E}\{\sum\limits_{d\in\mathcal{D}}\sum\limits_{r\in\mathcal{R}} \lambda_r^d(t)\cdot(Vs_d + Vb_r^d + V\alpha L_r^d + H_d(t))|\Theta(t)\} \end{array}, \tag{19}$$

where $B = \frac{1}{2}\sum\limits_{d\in\mathcal{D}}\{2(\sum\limits_{k\in\mathcal{K}} n_d^{k,\max}v_k)^2 + (\varepsilon_d)^2 + (\sum\limits_{r\in\mathcal{R}} A_{\max}^r)\}$. Detailed proofs please see the Appendix A.

4.2. **Online control algorithm design.** Fortunately, a careful investigation of the R.H.S of inequality (19) reveals that the optimization problem can be equivalently decoupled into two subproblems: 1) data allocation and 2) resource provisioning. The details of solving the two subproblems are presented as follows.

1) Data Allocation: To minimize the R.H.S of (19), by observing the relationship among variables, the part related to Data Allocation can be extracted from the R.H.S of (19) as:

$$\mathbb{E}\{\sum_{d\in\mathcal{D}}\sum_{r\in\mathcal{R}} \lambda_r^d(t)\cdot(Vs_d + Vb_r^d + V\alpha L_r^d + H_d(t))|\Theta(t)\}. \tag{20}$$

Furthermore, since the data generated from each location are independent, the centralized minimization can be implemented independently and distributedly. Considering the data allocation in location r at time t, we should solve the following problem.

$$\begin{array}{l} \min \sum\limits_{d\in\mathcal{D}} \lambda_r^d(t)[Vs_d + Vb_r^d + V\alpha L_r^d + H_d(t)] \\ s.t (9)(10) \end{array}. \tag{21}$$

In fact, the above problem is a generalized min-weight problem where the amount of data from location r moved to datacenter d $\lambda_r^d(t)$ is weighted by the queue backlog $H_d(t)$, bandwidth price b_d, storage price s_d and the latency cost $L(r,d)$. By using linear program theory (e.g., Simplex Method), we can get the following solution:

$$\lambda_r^d(t) = \begin{cases} a_r(t) & \text{d} = \text{d}^* \\ 0 & \text{else} \end{cases}, \tag{22}$$

where $\text{d}^* = \min_d[Vs_d + Vb_r^d + V\alpha L_r^d + H_d(t)]$. Obviously, the solution exhibits that the data generated from location r will incline to be moved to the datacenter with

the shortest weighted workload queue and the minimal operation price (e.g., VM price, Storage price etc.) at current time slot.

2) Resource Provisioning: The left part of R.H.S (19) related to variable $n_d^k(t)$ can be considered as resource provisioning problem if we remove the constant term. Therefore, we can get the optimal VM provisioning strategy by solving the following problem:

$$\begin{array}{l} \min\ \mathbb{E}\{\sum\limits_{d\in\mathcal{D}}\sum\limits_{k\in\mathcal{K}} n_d^k(t)\cdot(Vp_d^k(t)-H_d(t)v_k-Z_d(t)v_k)|\Theta(t)\} \\ s.t(11) \end{array}. \tag{23}$$

Since the resource provisioning problems in each datacenter are independent, similar to data allocation problem, (23) can be solved distributedly within each datacenter. For a single datacenter d, the resource provisioning problem can be further rewritten as (24).

$$\begin{array}{l} \min \mathbb{E}\{\sum\limits_{k\in\mathcal{K}} n_d^k(t)\cdot(Vp_d^k(t)-H_d(t)v_k-Z_d(t)v_k)|\Theta(t)\} \\ s.t(11) \end{array}. \tag{24}$$

The optimal solution to the above linear problem is:

$$n_d^k(t)=\begin{cases} n_d^{k,\max}, \text{if } H_d(t)+Z_d(t)>\frac{Vp_d^k(t)}{v_k} \\ 0, \text{ if } H_d(t)+Z_d(t)\le\frac{Vp_d^k(t)}{v_k} \end{cases}. \tag{25}$$

The above solution indicates that a type-k is preferred to be rented in t when its price , $p_d^k(t)$, is small, and VMs whose capacity, v_k, is large are more likely to be rented too.

Obviously, the two complex problems of data allocation and resource provisioning have been solved efficiently by using Lyapunov framework so far. The simple strategy facilitates the online deployment of the algorithm in the real world systems. The detail of its online algorithm is presented in Algorithm 1.

Algorithm 1: Procedures of the Proposed online Algorithm in Time Slot t

Input:
$H_d(t), Z_d(t), a_r(t), v_k, s_d, b_r^d, L_r^d, n_d^{k,\max}, A_{\max}^r, p_d^k(t), V, \alpha$ $(\forall d\in\mathcal{D}, \forall r\in\mathcal{R}, \forall k\in\mathcal{K})$
Output:
$n_d^k(t), \lambda_r^d(t)$ $(\forall d\in\mathcal{D}, \forall r\in\mathcal{R}, \forall k\in\mathcal{K})$
Resource provisioning:
foreach *datacenter* $d\in\mathcal{D}$ **do**
    Getting the VM provisioning strategy $(n_d^k(t))$ by solving the problem (23) using (25);
Data Allocation:
foreach $r\in\mathcal{R}$ **do**
    Getting the data allocation strategy $\lambda_r^d(t)$ by solving the problem (20) using (22);
Update the queues $H_d(t), Z_d(t)$ according to queue dynamic equation (12)(13) respectively.

5. **Performance analysis.** Next, to show its priority, we analyze theoretically the performance of the algorithm 1 in terms of cost optimality , queueing delay bound, and the worst delay of data processing.

Theorem 5.1. ***(Cost Optimality)*** *Suppose the data generation rate $a_r(t), \forall r \in \mathcal{R}$ is identical independent distribution over time slots, for any control parameter $V > 0$, the algorithm can achieve a time average cost related with the optimal one as follows.*

$$\limsup_{T\to\infty} \frac{1}{T} \cdot \sum_{t=0}^{T-1} \mathbb{E}\{C(t)\} \leq C^* + \frac{B}{V}, \tag{26}$$

where C^* is the infimum of the time average cost when choosing the optimal control action, representing the theoretically optimal solution to the optimization, B is the same as defined in (19).

Proof. Please see the Appendix B. □

This theorem exhibits that the gap between the time average cost obtained by the algorithm proposed in this paper and the optimal cost obtained offline is with $\mathcal{O}(1/V)$. In particular, by choosing the control variable V, the time-average cost O is arbitrarily close to the optimal cost C^*.

Theorem 5.2. ***(Queues Delay Bound)*** *Assume ε_d satisfies $\varepsilon_d < \sum_{k\in\mathcal{K}} n_d^{k,\max} \cdot v_k$. Let $H_d^{\max}$ and $Z_d^{\max}$ be the upper bound of queue $H_d(t)$ and virtual queue $Z_d(t)$ respectively. We have:*

$$Z_d^{\max} = \frac{V p_d^{\max}}{v_{\min}} + \varepsilon_d, \tag{27}$$

and

$$H_d^{\max} = \frac{V p_d^{\max}}{v_{\min}} + \sum_{r\in\mathcal{R}} A_{\max}^r \tag{28}$$

where $p_d^{\max}$ is the max price for each type of VM over time slots, and $v_{\min}$ is the minimal capacity among all kinds of VMs.

Proof. Please see the Appendix C. □

This theorem shows that the queue backlog is with $\mathcal{O}(V)$. It means that, to keep the queue backlog stable, we should choose a small V. Notice that decreasing V will cause a larger cost as shown in (26), the cost and system stability has an $[\mathcal{O}(1/V), \mathcal{O}(V)]$ tradeoff. In reality, given the acceptable cost we can choose the V to maximize the system stability, and vice versa.

Theorem 5.3. ***(Worst Case Delay)****Assume that the system running in First-in-First-Out manner, then the worst delay of the data processing in queue d is bounded by the l defined below:*

$$l = [H_d^{\max} + Z_d^{\max}/\varepsilon_d], \tag{29}$$

where $[x]$ denotes the minimal integer among those greater or equal to x and $H_d^{\max}$, $Z_d^{\max}$ are defined in (28) and (27).

Proof. Please see the Appendix D. □

This implies that the data arriving at any time slot t in Q_d can be completed within l time slots, demonstrating that our algorithm is able to guarantee the QoS (Quality of Service) for DSP. In addition, as can be seen, given the system parameters, by choosing a suitable ε_d, the QoS for the DSP can be changed. Also, with different setting of ε_d for $d \in \mathcal{D}$, we can achieve heterogeneous QoS for different datacenters.

6. **Conclusion.** Targeting the processing of big data from different locations in geo-distributed datacenters, we propose a systematical way for data moving and resource provisioning with the goal of cost minimization. The model takes into consideration the case that data analysis application is running in dynamic environment (e.g., unpredictable data generation, dynamic VM pricing). By using the Lyapunov technique, we transformed the original problem into two independent subproblems that can be solved efficiently online. Theoretical analysis demonstrates that the algorithm is able to maintain the stability of the dynamic system and complete the data processing within some time slots. It remains to further validate the effectiveness of the proposed algorithm via extensive experiments. Other considerations that may be further incorporated into our proposed framework include data processing relevance between two consecutive time slots, data processing migration among datacenters etc.

Appendix A. **Derivation of bound for the Drift-plus-penalty.** For the vectors of $\Theta(t) = [\mathbf{Z}(t); \mathbf{H}(t)]$, note that $(\max[x-y,0]+z)^2 \leq x^2+y^2+z^2+x(z-y)$, then we have:

$$\begin{aligned}
&Z_d(t+1)^2 - Z_d(t)^2 \\
&\leq \{1_{(H(t)>0)}(\varepsilon_d - \sum_{k\in\mathcal{K}} n_d^k(t)v_k) - 1_{(H(t)=0)} \sum_{k\in\mathcal{K}} n_d^{k,\max} v_k\} \\
&+2Z_d(t)\{1_{(H(t)>0)}(\varepsilon_d - \sum_{k\in\mathcal{K}} n_d^k(t)v_k) - 1_{(H(t)=0)} \sum_{k\in\mathcal{K}} n_d^{k,\max} v_k\} \\
&\leq (\varepsilon_d)^2 + (\sum_{k\in\mathcal{K}} n_d^{k,\max} v_k)^2 + 2Z_d(t)(\varepsilon_d - \sum_{k\in\mathcal{K}} n_d^k(t)v_k)
\end{aligned} \tag{30}$$

$$\begin{aligned}
&H_d(t+1)^2 - H_d(t)^2 \\
&\leq (\sum_{k\in\mathcal{K}} n_d^k(t)v_k)^2 + (\sum_{r\in\mathcal{R}} \lambda_r^d(t))^2 + 2H_d(t)(\sum_{r\in\mathcal{R}} \lambda_r^d(t) - \sum_{k\in\mathcal{K}} n_d^k(t)v_k)
\end{aligned} \tag{31}$$

Since $\lambda_r^d(t)$, $n_d^k(\tau)$ are bound by $A_{\max}^r$, $n_d^{k,\max}$ respectively. By defining $B = \frac{1}{2}\sum_{d\in\mathcal{D}}\{2(\sum_{k\in\mathcal{K}} n_d^{k,\max} v_k)^2 + (\varepsilon_d)^2 + (\sum_{r\in\mathcal{R}} A_{\max}^r)\}$, we can get the 1-slot Lyapunov drift as follows:

$$\begin{aligned}
&\Delta(\Theta(t)) \\
&= \tfrac{1}{2}\sum_{d\in\mathcal{D}} \mathbb{E}\{H_d(t+1)^2 - H_d(t)^2 | \Theta(t)\} \\
&+\tfrac{1}{2}\sum_{d\in\mathcal{D}} \mathbb{E}\{Z_d(t+1)^2 - Z_d(t)^2 | \Theta(t)\} \\
&\leq B + \sum_{d\in\mathcal{D}} \mathbb{E}\{H_d(t)(\sum_{r\in\mathcal{R}} \lambda_r^d(t) - \sum_{k\in\mathcal{K}} n_d^k(t)v_k) | \Theta(t)\} \\
&+ \sum_{d\in\mathcal{D}} \mathbb{E}\{Z_d(t)(\varepsilon_d - \sum_{k\in\mathcal{K}} n_d^k(t)v_k) | \Theta(t)\}
\end{aligned} \tag{32}$$

So far, by adding the term $V \cdot \mathbb{E}\{C(t)\}$ to the above expression, we can get the drift-plus-penalty bound in (19).

Appendix B. **Proof of Theorem 5.1.** To prove this theorem, we first give the following lemma.

Lemma B.1. *(Existence of Optimal Randomized Stationary Policy): There exists at least one policy π that chooses feasible solution $(n_d^{k,\pi}(t), \lambda_r^{d,\pi}(t)$ for $\forall d \in \mathcal{D}, \forall r \in$*

$\mathcal{R}, \forall k \in \mathcal{K}, \forall t \in [1, T]$*) and satisfies that:*

$$\begin{aligned} &\mathbb{E}\{C(t)\} = C^*, \\ &\mathbb{E}\{\sum_{r\in\mathcal{R}} \lambda_r^{d,\pi}(t)\} \leq \mathbb{E}\{\sum_{k\in\mathcal{K}} n_d^{k,\pi}(t) v_k\}, \\ &\varepsilon_d \leq \mathbb{E}\{\sum_{k\in\mathcal{K}} n_d^{k,\pi}(t) v_k\} \end{aligned} \tag{33}$$

where C^* is the theoretical lower bound of the cost. This lemma can be proved by using Caratheodory's theorem in [8].

Based on lemma B.1, next we prove the time averaged cost bound of our algorithm in (26) as follows.

Proof. From lemma B.1, it can be derived that there exist a constant $\delta > 0$ that satisfies:

$$\mathbb{E}\{\sum_{r\in\mathcal{R}} \lambda_r^{d,\pi}(t)\} \leq \mathbb{E}\{\sum_{k\in\mathcal{K}} n_d^{k,\pi}(t) v_k\} - \delta \tag{34}$$

$$\varepsilon_d \leq \mathbb{E}\{\sum_{k\in\mathcal{K}} n_d^{k,\pi}(t) v_k\} - \delta \tag{35}$$

Therefore, recall that our algorithm seek to minimize the right-hand-side of the inequality in (19) by choosing the decision variables among all feasible decisions at each time slot and apply lemma B.1, (34) and (35) into (19), we can obtain:

$$\begin{aligned} &\Delta(\Theta(t)) + V \cdot \mathbb{E}\{C(t)|\Theta(t)\} \\ &\leq B + VC^* - \delta \sum_{d\in\mathcal{D}} \mathbb{E}\{H_d(t)\} - \delta \sum_{d\in\mathcal{D}} \mathbb{E}\{Z_d(t)\} \end{aligned} \tag{36}$$

Taking the expectation for (36) and using the fact that $\Delta(\Theta(t)) = \mathbb{E}\{L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)\}$, it can be given that:

$$\begin{aligned} &\mathbb{E}\{L(\Theta(t+1)) - L(\Theta(t))\} + V \cdot \mathbb{E}\{C(t)\} \\ &\leq B + VC^* - \delta \sum_{d\in\mathcal{D}} \mathbb{E}\{H_d(t)\} - \delta \sum_{d\in\mathcal{D}} \mathbb{E}\{Z_d(t)\} \end{aligned} \tag{37}$$

With the law of telescoping sums over $t = 0, ..., T-1$ and then dividing the result by T gives:

$$\begin{aligned} &\tfrac{\mathbb{E}\{L(\Theta(T)) - L(\Theta(0))\}}{T} + \tfrac{V}{T} \cdot \sum_{t=0}^{T-1} \mathbb{E}\{C(t)\} \\ &\leq B + VC^* - \tfrac{\delta}{T} \sum_{t=0}^{T-1} \sum_{d\in\mathcal{D}} \mathbb{E}\{H_d(t)\} - \tfrac{\delta}{T} \sum_{t=0}^{T-1} \sum_{d\in\mathcal{D}} \mathbb{E}\{Z_d(t)\} \end{aligned} \tag{38}$$

Rearranging the terms and considering the fact that $L(\Theta(0))\} = 0, H_d(t) \geq 0, Z_d(t) \geq 0$, we obtain:

$$\frac{1}{T} \cdot \sum_{t=0}^{T-1} \mathbb{E}\{C(t)\} \leq C^* + \frac{B}{V} \tag{39}$$

Now (26) follows by taking a limit as $T \to \infty$. □

Appendix C. Proof of Theorem 5.2.

Proof. For $Z_d(t)$, as known to us,$Z_d(0) = 0 < Z_d^{\max}$. For any time slot $t \in [0, T]$, if $Z_d(t) \leq \frac{Vp_d^{\max}}{v_{\min}}$, then $Z_d(t+1) \leq \frac{Vp_d^{\max}}{v_{\min}} + \varepsilon_d$ (because $Z_d(t+1)$ will increase at most ε_d as shown in (13)). If $Z_d(t) > \frac{Vp_d^{\max}}{v_{\min}}$, then $H_d(t) + Z_d(t) > \frac{Vp_d^{\max}}{v_{\min}} > \frac{Vp_d^k}{v_k}$, and the queue will decrease by $\sum_{k\in\mathcal{K}} n_d^{k,\max} \cdot v_k$ (refer to (25)). Note that $\varepsilon_d < \sum_{k\in\mathcal{K}} n_d^{k,\max} \cdot v_k$, hence, $Z_d(t+1) < Z_d(t) \leq Z_d^{\max}$. The bound of queue $Z_d(t)$ is proved.

Similarly, for $H_d(t)$, $H_d(0) = 0 < H_d^{\max}$. For any time slot t, if $H_d(t) \leq \frac{Vp_d^{\max}}{v_{\min}}$, then $H_d(t+1) \leq \frac{Vp_d^{\max}}{v_{\min}} + \sum_{r\in\mathcal{R}} A_{\max}^r$ (because $H_d(t+1)$ will increase at most $\sum_{r\in\mathcal{R}} A_{\max}^r$ as shown in (12)). If $H_d(t) > \frac{Vp_d^{\max}}{v_{\min}}$, then $H_d(t) + Z_d(t) > \frac{Vp_d^{\max}}{v_{\min}} \geq \frac{Vp_d^k}{v_k}$ and the queue will decrease by $\sum_{k\in\mathcal{K}} n_d^{k,\max} \cdot v_k$ (refer to (25)). Noting that $\sum_{r\in\mathcal{R}} A_{\max}^r \leq \sum_{k\in\mathcal{K}} n_d^{k,\max} \cdot v_k$, so that the queue cannot increase on the next slot, i.e., $H_d(t+1) < H_d(t) \leq H_d^{\max}$. Hence, the bound of queue $H_d(t)$ is proved. □

Appendix D. Proof of Theorem 5.3.

Proof. If there is a time slot $\tau \in [t+1, t+l]$ satisfies that $H_d(\tau) = 0$, then the data arriving at time slot t are processed within l time slots. For the case that $H_d(\tau) > 0$ for any $\tau \in [t+1, t+l]$, the virtual queue has the departure rate $n_d^k(\tau)$ as show in (13). Therefore, we have:

$$\begin{aligned} Z_d(\tau+1) &= \max[Z_d(\tau) + \varepsilon_d - \sum_{k\in\mathcal{K}} n_d^k(t) \cdot v_k, 0] \\ &\geq Z_d(\tau) + \varepsilon_d - \sum_{k\in\mathcal{K}} n_d^k(t) \cdot v_k \end{aligned} \tag{40}$$

Summing the above inequality over time slots $\tau \in [t+1, t+l]$, we have:

$$Z_d(t+l+1) \geq Z_d(t+1) + l_d \cdot \varepsilon_d - \sum_{\tau=t+1}^{t+l+1} \sum_{k\in\mathcal{K}} n_d^k(\tau) \cdot v_k. \tag{41}$$

Hence, it can be derived that $\sum_{\tau=t+1}^{t+l+1} \sum_{k\in\mathcal{K}} n_d^k(\tau) \cdot v_k \geq l \cdot \varepsilon_d - Z_d^{\max}$. Note $l \cdot \varepsilon_d - Z_d^{\max} \geq H_d^{\max}$ when $l = [H_d^{\max} + Z_d^{\max}/\varepsilon_d]$, we have:

$$\sum_{\tau=t+1}^{t+l+1} \sum_{k\in\mathcal{K}} n_d^k(\tau) \cdot v_k \geq H_d^{\max} \geq H_d(t). \tag{42}$$

Since the system running in First-in-First-out model, the data arriving at time slot t will be processed before that arrives after time slot t. As the total amount of data processed within l time slots surpass the $H_d(t)$, then all the data arriving at time slot t will be served within l time slots, thus the worst delay is l time slots. □

REFERENCES

[1] Moving an elephant: Large scale hadoop data migration at facebook, http://www.facebook.com/notes/paul-yang/moving-an-elephant-large-scale-hadoop-data-migration-at-facebook/10150246275318920.

[2] AWS Import/Export, http://aws.amazon.com/importexport/.

[3] P. Barham, B. Dragovic and K. Fraser, Xen and the art of virtualization, *SIGOPS Operating Systems Review*, **37** (2003), 164–177.
[4] B. Cho and I. Gupta, New algorithms for planning bulk transfer via internet and shipping networks, in *Proc. IEEE ICDCS*, (2010), 305–314.
[5] B. Cho and I. Gupta, Budget-constrained bulk data transfer via internet and shipping networks, in *Proc. ACM ICAC*, (2011), 71–80.
[6] J. Dean and S. Ghemawat, MapReduce: Simplified data processing on large clusters, *Communications of the ACM*, **51** (2008), 107–113.
[7] Y. Feng, B. Li and B. Li, Airlift: Video conferencing as a cloud service using inter-datacenter networks, in *Proceedings of the IEEE International Conference on Network Protocols(ICNP'12)*, (2012), 1–11.
[8] L. Georgiadis, M. J. Neely and L. Tassiulas, Resource allocation and cross-layer control in wireless networks, *Foundations and Trends in Networking,* **1** (2006), 1–144.
[9] Z. Huang, C. Mei, L. Li and T. Woo, CloudStream: Delivering high-quality streaming videos through a cloud-based SVC proxy, in *Proceedings of the IEEE INFOCOM*, (2011), 201–205.
[10] 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*, **25** (2014), 2648–2658.
[11] X. Mo and H. Wang, Asynchronous index strategy for high performance real-time big data stream storage, *in Network Infrastructure and Digital Content (IC-NIDC)*, (2012), 232–236.
[12] X. Nan, Y. He and L. Guan, Optimal resource allocation for multimedia cloud based on queuing model, in *Proc. of IEEE MMSP Workshop*, (2011), 1–6.
[13] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*, Morgan and Claypool, 2010.
[14] M. J. Neely, Opportunistic scheduling with worst case delay guarantees in single and multi-hop networks, in *Proc. of INFOCOM*, (2011), 1728–1736.
[15] E. E. Schadt, M. D. Linderman, J. Sorenson, L. Lee and G. P. Nolan, Computational solutions to large-scale data management and analysis, *Nat Rev Genet*, **11** (2010), 647–657.
[16] J. Tang, W. P. Tay and Y. Wen, Dynamic request redirection and elastic service scaling in cloud-centric media networks, *IEEE Transactions on Multimedia*, **16** (2014), 1434–1445.
[17] L. Tassiulas and A. Ephremides, Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks, IEEE Transactions on Automatic Control, **37** (1992), 1936–1948.
[18] C. Union, Homepage `http://www.cloudunion.cn/`.
[19] R. Urgaonkar, U. Kozat, K. Igarashi and M. J. Neely, Resource allocation and power management in virtualized data centers, in *Proceedings of the IEEE Network Operations and Management Symp(NOMS'10)*, (2010), 479–486.
[20] J. Wang, W. Bao, X. Zhu, L. T. Yang and Y. Xiang, FESTAL: Fault-tolerant elastic scheduling algorithm for real-time tasks in virtualized clouds, *IEEE Transactions on Computers*, **64** (2014), 2445–2558.
[21] F. Wang, J. Liu and M. Chen, CALMS: Cloud-assisted live media streaming for globalized demands with time/ region diversities, in *Proceedings of the IEEE INFOCOM*, (2012), 199–207.
[22] D. Wu, Z. Xue and J. He, iCloudAccess: Cost-effective streaming of videogames from the cloud with low latency, *IEEE Transactions on Circuits and Systems for Video Technology*, **28** (2014), 1405–1416.
[23] Y. Wu, C. Wu, B. Li, X. Qiu and F.C.M. Lau , Cloudmedia: When cloud on demand meets video on demand, In *Proc. of IEEE ICDCS*, (2011), 268–277.
[24] Y. Wu, C. Wu, B. Li, L. Zhang, Z. Li and F. Lau, Scaling social media applications into geo-distributed clouds, in *Proc. IEEE INFOCOM*, (2012), 684–692.
[25] W. Xiao, W. Bao, X. Zhu, C. Wang, L. Chen and L. T. Yang, Dynamic request redirection and resource provisioning for cloud-based video services under heterogeneous environment, *IEEE Transactions on Parallel and Distributed Systems*, **pp** (2015), p1.
[26] Y. Yao, L. Huang and A. B. Sharma, L. Golubchik and M. J. Neely, Power cost reduction in distributed data centers: A two-time-scale approach for delay tolerant workloads, *IEEE Transactions On Parallel and Distributed Systems*, **25** (2014), 200–211.
[27] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker and I. Stoica. Spark: cluster computing with working sets, In *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing(HotCloud'10)*, Berkeley, CA, USA, (2010), p10.

[28] L. Zhang, C. Wu, Z. Li, C. Guo, M. Chen and F. C. M. Lau, Moving big data to the cloud: An online cost-minimizing approach, *IEEE Journal on Selected Areas in Communications*, **31** (2013), 2710–2721.
[29] X. Zhu, C. Chen, L. T. Yang and Y. Xiang, ANGEL: Agent-based scheduling for real-time tasks in virtualized clouds, *IEEE Transactions on Computers*, **pp** (2015), p1.
[30] X. Zhu, R. Ge, J. Sun and C. He, 3E: Energy-efficient elastic scheduling for independent tasks in heterogeneous computing system, *Journal of Systems and Software*, **86** (2012), 302–314.
[31] X. Zhu, X. Qin and M. Qiu, QoS-aware fault-tolerant scheduling for real-time tasks on heterogeneous clusters, *IEEE Transactions on Computers*, **60** (2011), 800–812.

Received July 2015; revised August 2015.

E-mail address: wdbao@nudt.edu.cn
E-mail address: wenhuaxiao@nudt.edu.cn
E-mail address: hrji@nudt.edu.cn
E-mail address: chenchao@nudt.edu.cn
E-mail address: xmzhu@nudt.edu.cn
E-mail address: wujh@mathstat.yorku.ca