Received 15 November 2013; revised 20 February 2013; accepted 25 February 2014. Date of publication 10 March 2014; date of current version 30 October 2014.

Digital Object Identifier 10.1109/TETC.2014.2310456

Cost Minimization for Big Data Processing in Geo-Distributed Data Centers

LIN GU, (Student Member, IEEE), DEZE ZENG, (Member, IEEE), PENG LI, (Member, IEEE) AND SONG GUO, (Senior Member, IEEE)

University of Aizu, Fukushima 965-8580, Japan

CORRESPONDING AUTHOR: S. GUO (sguo@u-aizu.ac.jp)

ABSTRACT The explosive growth of demands on big data processing imposes a heavy burden on computation, storage, and communication in data centers, which hence incurs considerable operational expenditure to data center providers. Therefore, cost minimization has become an emergent issue for the upcoming big data era. Different from conventional cloud services, one of the main features of big data services is the tight coupling between data and computation as computation tasks can be conducted only when the corresponding data are available. As a result, three factors, i.e., task assignment, data placement, and data movement, deeply influence the operational expenditure of data centers. In this paper, we are motivated to study the cost minimization problem via a joint optimization of these three factors for big data services in geo-distributed data centers. To describe the task completion time with the consideration of both data transmission and computation, we propose a 2-D Markov chain and derive the average task completion time in closed-form. Furthermore, we model the problem as a mixed-integer nonlinear programming and propose an efficient solution to linearize it. The high efficiency of our proposal is validated by extensive simulation-based studies.

INDEX TERMS Big data, data flow, data placement, distributed data centers, cost minimization, task assignment.

I. INTRODUCTION

Data explosion in recent years leads to a rising demand for big data processing in modern data centers that are usually distributed at different geographic regions, e.g., Google's 13 data centers over 8 countries in 4 continents [1]. Big data analysis has shown its great potential in unearthing valuable insights of data to improve decision-making, minimize risk and develop new products and services. On the other hand, big data has already translated into big price due to its high demand on computation and communication resources [2]. Gartner predicts that by 2015, 71% of worldwide data center hardware spending will come from the big data processing, which will surpass \$126.2 billion. Therefore, it is imperative to study the cost minimization problem for big data processing in geo-distributed data centers.

Many efforts have been made to lower the computation or communication cost of data centers. Data center resizing (DCR) has been proposed to reduce the computation cost by adjusting the number of activated servers via task placement [3]. Based on DCR, some studies have explored the geographical distribution nature of data centers and electricity price heterogeneity to lower the electricity cost [4]–[6]. Big data service frameworks, e.g., [7], comprise a distributed file system underneath, which distributes data chunks and their replicas across the data centers for fine-grained load-balancing and high parallel data access performance. To reduce the communication cost, a few recent studies make efforts to improve data locality by placing jobs on the servers where the input data reside to avoid remote data loading [7], [8].

Although the above solutions have obtained some positive results, they are far from achieving the cost-efficient big data processing because of the following weaknesses. First, data locality may result in a waste of resources. For example, most computation resource of a server with less popular data may stay idle. The low resource utility further causes more servers to be activated and hence higher operating cost.

Second, the links in networks vary on the transmission rates and costs according to their unique features [9], e.g., the distances and physical optical fiber facilities between data centers. However, the existing routing strategy among data centers fails to exploit the link diversity of data center networks. Due to the storage and computation capacity constraints, not all tasks can be placed onto the same server, on which their corresponding data reside. It is unavoidable that certain data must be downloaded from a remote server. In this case, routing strategy matters on the transmission cost. As indicated by Jin et al. [10], the transmission cost, e.g., energy, nearly proportional to the number of network link used. The more link used, the higher cost will be incurred. Therefore, it is essential to lower the number of links used while satisfying all the transmission requirements.

Third, the Quality-of-Service (QoS) of big data tasks has not been considered in existing work. Similar to conventional cloud services, big data applications also exhibit Service-Level-Agreement (SLA) between a service provider and the requesters. To observe SLA, a certain level of QoS, usually in terms of task completion time, shall be guaranteed. The QoS of any cloud computing tasks is first determined by where they are placed and how many computation resources are allocated. Besides, the transmission rate is another influential factor since big data tasks are data-centric and the computation task cannot proceed until the corresponding data are available. Existing studies, e.g., [3], on general cloud computing tasks mainly focus on the computation capacity constraints, while ignoring the constraints of transmission rate.

To conquer above weaknesses, we study the cost minimization problem for big data processing via joint optimization of task assignment, data placement, and routing in geo-distributed data centers. Specifically, we consider the following issues in our joint optimization. Servers are equipped with limited storage and computation resources. Each data chunk has a storage requirement and will be required by big data tasks. The data placement and task assignment are transparent to the data users with guaranteed QoS. Our objective is to optimize the big data placement, task assignment, routing and DCR such that the overall computation and communication cost is minimized. Our main contributions are summarized as follows:

- To our best knowledge, we are the first to consider the cost minimization problem of big data processing with joint consideration of data placement, task assignment and data routing. To describe the rate-constrained computation and transmission in big data processing process, we propose a two-dimensional Markov chain and derive the expected task completion time in closed form.
- Based on the closed-form expression, we formulate the cost minimization problem in a form of mixed-integer nonlinear programming (MINLP) to answer the following questions: 1) how to place these data chunks in the servers, 2) how to distribute tasks onto servers without violating the resource constraints, and 3) how to resize data centers to achieve the operation cost minimization goal.
- To deal with the high computational complexity of solving MINLP, we linearize it as a mixed-integer linear programming (MILP) problem, which can be solved using commercial solver. Through extensive numerical studies, we show the high efficiency of our proposed joint-optimization based algorithm.

The rest of the paper is organized as follows. Section II summaries the related work. Section III introduces our system model. The cost optimization is formulated as an MINLP problem in Section IV and then it is linearized in Section V. The theoretical findings are verified by experiments in Section VI. Finally, Section VII concludes our work.

II. RELATED WORK

A. SERVER COST MINIMIZATION

Large-scale data centers have been deployed all over the world providing services to hundreds of thousands of users. According to [11], a data center may consist of large numbers of servers and consume megawatts of power. Millions of dollars on electricity cost have posed a heavy burden on the operating cost to data center providers. Therefore, reducing the electricity cost has received significant attention from both academia and industry [5], [11]–[13]. Among the mechanisms that have been proposed so far for data center energy management, the techniques that attract lots of attention are task placement and DCR.

DCR and task placement are usually jointly considered to match the computing requirement. Liu et al. [4] re-examine the same problem by taking network delay into consideration. Fan et al. [12] study power provisioning strategies on how much computing equipment can be safely and efficiently hosted within a given power budget. Rao et al. [3] investigate how to reduce electricity cost by routing user requests to geo-distributed data centers with accordingly updated sizes that match the requests. Recently, Gao et al. [14] propose the optimal workload control and balancing by taking account of latency, energy consumption and electricity prices. Liu et al. [15] reduce electricity cost and environmental impact using a holistic approach of workload balancing that integrates renewable supply, dynamic pricing, and cooling supply.

B. BIG DATA MANAGEMENT

To tackle the challenges of effectively managing big data, many proposals have been proposed to improve the storage and computation process.

The key issue in big data management is reliable and effective data placement. To achieve this goal, Sathiamoorthy et al. [16] present a novel family of erasure codes that are efficiently repairable and offer higher reliability compared to Reed-Solomon codes. They also analytically show that their codes are optimal on an identified tradeoff between locality and minimum distance. Yazd et al. [8] make use of flexibility in the data block placement policy to increase energy efficiency in data centers and propose a scheduling algorithm, which takes into account energy efficiency in addition to fairness and data locality properties. Hu et al. [17] propose a mechanism allowing linked open data to take advantage of existing large-scale data stores to meet the requirements on distributed and parallel data processing.

Moreover, how to allocate the computation resources to tasks has also drawn much attention. Cohen et al. [18] present new design philosophy, techniques and experience providing a new magnetic, agile and deep data analytics for one of the world's largest advertising networks at Fox Audience Network, using the Greenplum parallel database system. Kaushik et. al [19] propose a novel, data-centric algorithm to reduce energy costs and with the guarantee of thermal-reliability of the servers. Chen et al. [20] consider the problem of jointly scheduling all three phases, i.e., map, shuffle and reduce, of the MapReduce process and propose a practical heuristic to combat the high scheduling complexity.

C. DATA PLACEMENT

Shachnai et al. [21] investigate how to determine a placement of Video-on-Demand (VoD) file copies on the servers and the amount of load capacity assigned to each file copy so as to minimize the communication cost while ensuring the user experience. Agarwal et al. [22] propose an automated data placement mechanism Volley for geo-distributed cloud services with the consideration of WAN bandwidth cost, data center capacity limits, data inter-dependencies, etc. Cloud services make use of Volley by submitting logs of datacenter requests. Volley analyzes the logs using an iterative optimization algorithm based on data access patterns and client locations, and outputs migration recommendations back to the cloud service. Cidon et al. [23] invent MinCopysets, a data replication placement scheme that decouples data distribution and replication to improve the data durability properties in distributed data centers. Recently, Jin et al. [10] propose a joint optimization scheme that simultaneously optimizes virtual machine (VM) placement and network flow routing to maximize energy savings.

Existing work on data center cost optimization, big data management or data placement mainly focuses on one or two factors. To deal with big data processing in geo-distributed data centers, we argue that it is essential to jointly consider data placement, task assignment and data flow routing in a systematical way.

III. SYSTEM MODEL

In this section, we introduce the system model. For the convenience of the readers, the major notations used in this paper are listed in Table 1.

A. NETWORK MODEL

We consider a geo-distributed data center topology as shown in Fig. 1, in which all servers of the same data center (DC) are connected to their local switch, while data centers are connected through switches. There are a set I of data centers, and each data center $i \in I$ consists of a set J_i of servers that are connected to a switch $m_i \in M$ with a local transmission cost of C_L. In general, the transmission cost C_R for inter-data center traffic is greater than C_L, i.e., $C_R > C_L$. Without loss of generality, all servers in the network have the same computation resource and storage capacity, both of which are normalized to one unit. We use J to denote the set of all severs, i.e., $J = J_1 \bigcup J_2 \cdots \bigcup J_{|I|}$.

The whole system can be modeled as a directed graph $G = (N, E)$. The vertex set $N = M \bigcup J$ includes the set M of all switches and the set J of all servers, and E is the directional edge set. All servers are connected to, and only to, their local switch via intra-data center links while the switches are connected via inter-data center links determined by their physical connection. The weight of each link $w^{(u,v)}$, representing the corresponding communication cost, can be

TABLE 1. **Notations**

Constants	
J_i	The set of servers in data center i
m_i	The switch in data center i
$w^{(u,v)}$	The weight of link (u, v)
ϕ_k	The size of chunk k
λ_k	The task arrival rate for data chunk k
P	The number of data chunk replicas
D	The maximum expected response time
P_j	The power consumption of server j
$\gamma^{(u,v)}$	The transmission rate of link (u, v)
Variables	
x_j	A binary variable indicating if server j is activated or not
y_{jk}	A binary variable indicating if chunk k is placed on server j or not
$z_{jk}^{(u,v)}$	A binary variable indicating if link (u, v) is used for flow for chunk k on server j
λ_{jk}	The request rate for chunk k on server j
θ_{jk}	The CPU usage of chunk k on server j
μ_{jk}	The CPU processing rate of chunk k on server j
$f_{jk}^{(u,v)}$	The flow for chunk k destined to server j through link (u, v)

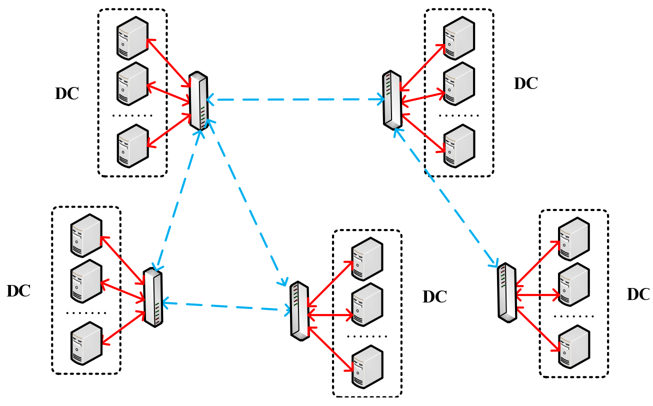


FIGURE 1. **Data center topology.**

defined as

$$w^{(u,v)} = \begin{cases} C_R, & \text{if } u, v \in M, \\ C_L, & \text{otherwise.} \end{cases} \tag{1}$$

B. TASK MODEL

We consider big data tasks targeting on data stored in a distributed file system that is built on geo-distributed data centers. The data are divided into a set K of chunks. Each chunk $k \in K$ has the size of $\phi_k(\phi_k \leq 1)$, which is normalized to the server storage capacity. P-way replica [19] is used in our model. That is, for each chunk, there are exactly P copies stored in the distributed file system for resiliency and fault-tolerance.

It has been widely agreed that the tasks arrival at data centers during a time period can be viewed as a Poisson process [9], [24]. In particular, let λ_k be the average task arrival rate requesting chunk k. Since these tasks will be distributed to servers with a fixed probability, the task arrival in each server can be also regarded as a Poisson process. We denote the average arrival rate of task for chunk k on server j as $\lambda_{jk}(\lambda_{jk} \leq 1)$. When a task is distributed to a server where its requested data chunk does not reside, it needs to wait for the data chunk to be transferred. Each task should be responded in time D.

Moreover, in practical data center management, many task predication mechanisms based on the historical statistics have been developed and applied to the decision making in data centers [19]. To keep the data center settings up-to-date, data center operators may make adjustment according to the task predication period by period [3], [14], [15]. This approach is also adopted in this paper.

IV. PROBLEM FORMULATION

In this section, we first present the constraints of data and task placement, remote data loading, and QoS. Then, we give the complete formulation of the cost minimization problem in a mixed-integer nonlinear programming form.

A. CONSTRAINTS OF DATA AND TASK PLACEMENT

We define a binary variable y_{jk} to denote whether chunk k is placed on server j as follows,

$$y_{jk} = \begin{cases} 1, & \text{if chunk } k \text{ is placed on server } j, \\ 0, & \text{Otherwise.} \end{cases} \tag{2}$$

In the distributed file system, we maintain P copies for each chunk $k \in K$, which leads to the following constraint:

$$\sum_{j \in J} y_{jk} = P, \forall k \in K. \tag{3}$$

Furthermore, the data stored in each server $j \in J$ cannot exceed its storage capacity, i.e.,

$$\sum_{k \in K} y_{jk} \cdot \phi_k \leq 1, \forall j \in J. \tag{4}$$

As for task distribution, the sum rates of task assigned to each server should be equal to the overall rate,

$$\lambda_k = \sum_{j \in J} \lambda_{jk}, \forall k \in K. \tag{5}$$

Finally, we define a binary variable x_j to denote whether server j is activated, i.e.,

$$x_j = \begin{cases} 1, & \text{if this server is activated,} \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

A server shall be activated if there are data chunks placed onto it or tasks assigned to it. Therefore, we have

$$\frac{\sum_{k \in K} y_{jk} + \sum_{k \in K} \lambda_{jk}}{K + \sum_{k \in K} \lambda_k} \leq x_j \leq \sum_{k \in K} y_{jk} + A \sum_{k \in K} \lambda_{jk}, \forall j \in J, \tag{7}$$

where A is an arbitrarily large number.

B. CONSTRAINTS OF DATA LOADING

Note that when a data chunk k is required by a server j, it may cause internal and external data transmissions. This routing procedure can be formulated by a flow model. All the nodes N in graph G, including the servers and switches, can be divided into three categories:

- Source nodes $u(u \in J)$. They are the servers with chunk k stored in it. In this case, the total outlet flows to destination server j for chunk k from all source nodes shall meet the total chunk requirement per time unit as $\lambda_{jk} \cdot \phi_k$.
- Relay nodes $m_i(m_i \in M)$. They receive data flows from source nodes and forward them according to the routing strategy.
- Destination node $j(j \in J)$. When the required chunk is not stored in the destination node, i.e., $y_{jk} = 0$, it must receive the data flows of chunk k at a rate $\lambda_{jk} \cdot \phi_k$.

Let $f_{jk}^{(u,v)}$ denote the flow over the link $(u, v) \in E$ carrying data of chunk $k \in K$ and destined to server $j \in J$. Then, the constraints on the above three categories of nodes can be expressed as follows respectively.

$$f_{jk}^{(u,v)} \leq y_{uk} \cdot \lambda_k \cdot \phi_k, \forall (u, v) \in E, u, j \in J, k \in K \tag{8}$$

$$\sum_{(u,v) \in E} f_{jk}^{(u,v)} - \sum_{(v,w) \in E} f_{jk}^{(v,w)} = 0, \forall v \in M, j \in J, k \in K \tag{9}$$

$$\sum_{(u,j) \in E} f_{jk}^{(u,j)} = (1 - y_{jk})\lambda_{jk} \cdot \phi_k, \forall j \in J, k \in K \tag{10}$$

Note that a non-zero flow $f_{jk}^{(u,v)}$ emitting from server u only if it keeps a copy of chunk k, i.e., $y_{uk} = 1$, as characterized in (8). The flow conservation is maintained on each switch as shown in (9). Finally, the destination receives all data $\lambda_k \cdot \phi_k$ from others only when it does not hold a copy of chunk k, i.e., $y_{ik} = 0$. This is guaranteed by (10).

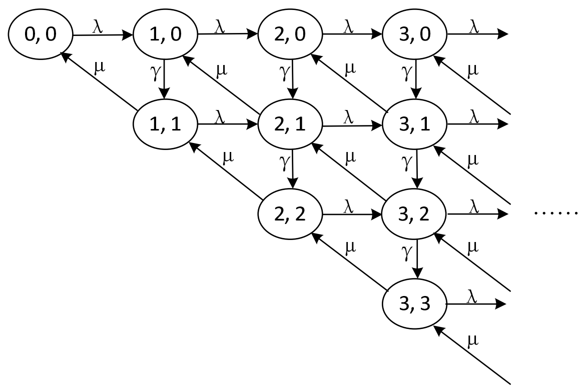


FIGURE 2. **Two-dimensional Markov Chain.**

C. CONSTRAINTS OF QoS SATISFACTION

Let μ_{jk} and γ_{jk} be the processing rate and loading rate for data chunk k on server j, respectively. The processing procedure then can be described by a two-dimensional Markov chain as Fig. 2, where each state (p, q) represents p pending tasks and q available data chunks.

We let θ_{jk} denote the amount of computation resource (e.g., CPU) that chunk k occupies. The processing rate of tasks is proportional to its computation resource usage, i.e.,

$$\mu_{jk} = \alpha_j \cdot \theta_{jk}, \forall j \in J, k \in K, \tag{11}$$

where α_j is a constant relying on the speed of server j.

Furthermore, the total computation resource allocated to all chunks on each server j shall not exceed its total computation resource, i.e.,

$$\sum_{k \in K} \theta_{jk} \leq 1, \forall j \in J. \tag{12}$$

The loading rate of γ_{jk} is constrained by the rate on any link (u, v) denoted as $\gamma_{(u,v)}$, if a non-zero flow $f_{jk}^{(u,v)}$ goes through it. This condition can be described by a binary variable $z_{jk}^{(u,v)}$ as

$$f_{jk}^{(u,v)} \leq z_{jk}^{(u,v)} \leq A f_{jk}^{(u,v)}, \forall (u, v) \in E, j \in J, k \in K. \tag{13}$$

Finally, the constraints on γ_{jk} is given as

$$\gamma_{jk} \leq \gamma^{(u,v)} \cdot z_{jk}^{(u,v)} + 1 - z_{jk}^{(u,v)}, \forall (u, v) \in E, j \in J, k \in K. \tag{14}$$

Note that we consider sufficient bandwidth on each link such that $\gamma^{(u,v)}$ can be handled as a constant number, which is mainly determined by I/O and switch latency [25].

By denoting $\pi_{jk}(p, q)$ as the steady state probability that the Markov chain stays at (p, q), we can describe the transition process by a group of ODEs as follows. According to the transition characteristics, the whole figure can be divided into three regions.

Region-I: all states in the first line. In Region-I, except state $(0, 0)$, state $(p, 0)(p > 1)$ transits to two neighboring states $(p + 1, 0)$ and $(p, 1)$. These can be described as:

$$\pi'_{jk}(0, 0) = -\lambda_{jk}\pi_{jk}(0, 0) + \mu_{jk}\pi_{jk}(1, 1), \forall j \in J, k \in K. \tag{15}$$

$$\begin{aligned}\pi'_{jk}(p, 0) &= -\lambda_{jk}(\pi_{jk}(p, 0) - \pi_{jk}(p - 1, 0)) \\ &\quad + \mu_{jk}\pi_{jk}(p + 1, 1) - \gamma\pi_{jk}(p, 0), \forall j \in J, k \in K.\end{aligned} \tag{16}$$

Region II: all states in the diagonal line except $(0, 0)$. In this region, all the pending tasks have already obtained their needed data chunk to proceed. Therefore, each state (p, q) in Region-II will transit to $(p-1, q-1)$ after processing one data chunk. Then, we have:

$$\begin{aligned}\pi'_{jk}(p, p) &= -\lambda_{jk}\pi_{jk}(p, p) + \mu_{jk}(\pi_{jk}(p + 1, p + 1) \\ &\quad -\pi_{jk}(p, p)) + \gamma\pi_{jk}(p, p-1), \forall j \in J, k \in K.\end{aligned} \tag{17}$$

Region-III: all remaining states in the central region. Each state (p, q) in Region-III relies on its three neighboring states and also will transit to the other three neighboring states. As shown in Fig. 2, the transition relationship can be written as:

$$\begin{aligned}\pi'_{jk}(p, q) &= -\lambda_{jk}(\pi_{jk}(p, q) - \pi_{jk}(p - 1, q - 1)) \\ &\quad + \mu_{jk}(\pi_{jk}(p + 1, q + 1) - \pi_{jk}(p, q)) \\ &\quad - \gamma(\pi_{jk}(p, q) - \pi_{jk}(p - 1, q - 1)), \\ &\quad \forall j \in J, k \in K.\end{aligned} \tag{18}$$

By solving the above ODEs, we can derive the state probability $\pi_{jk}(p, q)$ as:

$$\pi_{jk}(p, q) = \frac{(\lambda_{jk})^p(\mu_{jk})^{B-q}(\gamma_{jk})^{B-p+q}}{\sum_{q=0}^{B}\sum_{p=0}^{B}(\lambda_{jk})^p(\mu_{jk})^{B-q}(\gamma_{jk})^{B-p+q}}, \quad \forall j \in J, k \in K, \tag{19}$$

where B is the task buffer size on each server. When B goes to infinity, the mean number of tasks for chunk k on server j T_{jk} is

$$T_{jk} = \lim_{B \to \infty} \frac{\sum_{q=0}^{B}\sum_{p=0}^{B} p(\lambda_{jk})^p(\mu_{jk})^{B-q}(\gamma_{jk})^{B-p+q}}{\sum_{q=0}^{B}\sum_{p=0}^{B}(\lambda_{jk})^p(\mu_{jk})^{B-q}(\gamma_{jk})^{B-p+q}}, \quad \forall j \in J, k \in K. \tag{20}$$

By applying the multivariate l'Hospital's rule, (20) can be simplified to

$$T_{jk} = \frac{\lambda_{jk}}{\mu_{jk}\gamma_{jk} - \lambda_{jk}}, \forall j \in J, k \in K. \tag{21}$$

According to the Little's law, the expected delay d_{jk} of user requests for chunk k on server j is

$$d_{jk} = \frac{T_{jk}}{\lambda_{jk}} = \frac{1}{\mu_{jk}\gamma_{jk} - \lambda_{jk}}, \forall j \in J, k \in K. \tag{22}$$

According to the QoS requirement, i.e., $d_{jk} \leq D$, we have

$$\mu_{jk}\gamma_{jk} - \lambda_{jk} \geq \frac{u_{jk}}{D}, \forall j \in J, k \in K, \tag{23}$$

where

$$u_{jk} = \begin{cases} 1, \text{ if } \lambda_{jk} \neq 0, \\ 0, \text{ otherwise.} \end{cases} \tag{24}$$

The binary variable of u_{jk} can be described by constraints:

$$\lambda_{jk} \le u_{jk} \le A\lambda_{jk}, \forall j \in J, k \in K, \tag{25}$$

where A is an arbitrary large number, because of $0 < \lambda_{jk} < 1$ and $u_{jk} \in \{0, 1\}$.

D. AN MINLP FORMULATION

The total energy cost then can be calculated by summing up the cost on each server across all the geo-distributed data centers and the communication cost, i.e.,

$$C_{\text{total}} = \sum_{j\in J} x_j \cdot P_j + \sum_{j\in J}\sum_{k\in K}\sum_{(u,v)\in E} f_{jk}^{(u,v)} \cdot w^{(u,v)}, \tag{26}$$

where P_j is the cost of each activated server j.

Our goal is to minimize the total cost by choosing the best settings of $x_j, y_{jk}, z_{jk}^{(u,v)}, \theta_{jk}, \lambda_{jk}$ and $f_{jk}^{(u,v)}$. By summarizing all constraints discussed above, we can formulate this cost minimization as a mixed-integer nonlinear programming (MINLP) problem as:

$$\begin{aligned} &\textit{MINLP}: \\ &\quad \min: \quad (26), \\ &\quad \text{s.t.}: \quad (3)-(5), (7)-(14), (23), (25), \\ &\qquad\qquad x_j, y_{jk}, z_{jk}, u_{jk} \in \{0, 1\}, \forall j \in J, k \in K \end{aligned}$$

Note that the above formulation is based on the setting that the number of replicas for each data chunk is a predetermined constant. If it is a part of the optimization, denoted by an integer variable p, the total cost can be further minimized by the formulation below.

$$\begin{aligned} &\textit{MINLP}-2: \\ &\quad \min: \quad (26), \\ &\quad \text{s.t.}: \quad \sum_{j\in J} y_{jk} = p, \forall k \in K, \\ &\qquad\qquad p \ge 1, \\ &\qquad\qquad (4), (5), (7)-(14), (23), (25), \\ &\qquad\qquad x_j, y_{jk}, z_{jk}, u_{jk} \in \{0, 1\}, \forall j \in J, k \in K. \end{aligned}$$

V. LINEARIZATION

We observe that the constraints (8) and (10) are nonlinear due to the products of two variables. To linearize these constraints, we define a new variable δ_{jk} as follows:

$$\delta_{jk} = y_{jk}\lambda_{jk}, \forall j \in J, \forall k \in K, \tag{27}$$

which can be equivalently replaced by the following linear constraints:

$$0 \le \delta_{jk} \le \lambda_{jk}, \forall j \in J, \forall k \in K, \tag{28}$$

$$\lambda_{jk} + y_{jk} - 1 \le \delta_{jk} \le y_{jk}, \forall j \in J, \forall k \in K. \tag{29}$$

The constraints (8) and (10) can be written in a linear form as:

$$f_{jk}^{(u,v)} \le \delta_{uk}\phi_k, \forall (u, v) \in E, u, j \in J, k \in K, \tag{30}$$

$$\sum_{(u,j)\in E} f_{jk}^{(u,j)} = (\lambda_{jk} - \delta_{jk}) \cdot \phi_k, \forall j \in J, k \in K. \tag{31}$$

We then consider the remaining nonlinear constraints (14) and (23) that can be equivalently written as:

$$\gamma^{(u,v)}\mu_{jk}z_{jk}^{(u,v)} + 1 - \mu_{jk}z_{jk}^{(u,v)} - \lambda_{jk} \ge \frac{u_{jk}}{D}, \quad \forall (u, v) \in E, \forall j \in J, \forall k \in K. \tag{32}$$

In a similar way, we define a new variable ϵ_{jk} as:

$$\epsilon_{jk}^{(u,v)} = \mu_{jk}z_{jk}^{(u,v)}, \tag{33}$$

such that constraint (32) can be written as:

$$\gamma^{(u,v)}\epsilon_{jk}^{(u,v)} + \mu_{jk} - \epsilon_{jk}^{(u,v)} - \lambda_{jk} \ge \frac{u_{jk}}{D}, \quad \forall (u, v) \in E, \forall j \in J, \forall k \in K. \tag{34}$$

The constraint (33) can be linearized by:

$$0 \le \epsilon_{jk}^{(u,v)} \le \mu_{jk}, \forall (u, v) \in E, \forall j \in J, \forall k \in K, \tag{35}$$

$$\mu_{jk} + z_{jk}^{(u,v)} - 1 \le \epsilon_{jk}^{(u,v)} \le z_{jk}^{(u,v)}, \forall j \in J, \forall k \in K. \tag{36}$$

Now, we can linearize the **MINLP** problem into a mixed-integer linear programming (**MILP**) as

$$\begin{aligned} &\textit{MILP}: \\ &\quad \min: \quad (26), \\ &\quad \text{s.t.}: \quad (3)-(5), (7), (9), (11)-(13), \\ &\qquad\qquad (25), (28)-(31), (34)-(36), \\ &\qquad\qquad x_j, y_{jk}, z_{jk}, u_{jk} \in \{0, 1\}, \forall j \in J, k \in K. \end{aligned}$$

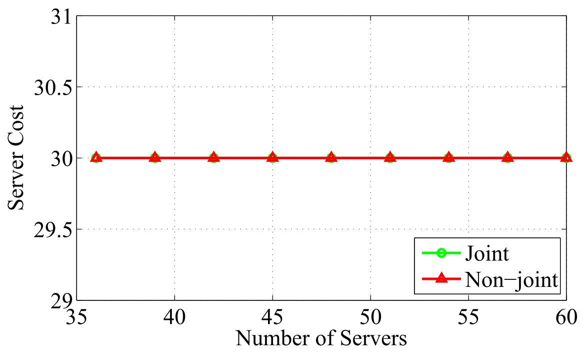
VI. PERFORMANCE EVALUATION

In this section, we present the performance results of our joint-optimization algorithm ("Joint") using the MILP formulation. We also compare it against a separate optimization scheme algorithm ("Non-joint"), which first finds a minimum number of servers to be activated and the traffic routing scheme using the network flow model as described in Section IV-B.

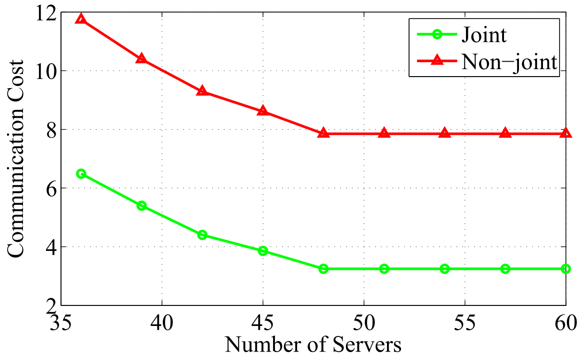
In our experiments, we consider $|J| = 3$ data centers, each of which is with the same number of servers. The intra- and inter-data center link communication cost are set as $C_L = 1$ and $C_R = 4$, respectively. The cost P_j on each activated server j is set to 1. The data size, storage requirement, and task arrival rate are all randomly generated. To solve the MILP problem, commercial solver Gurobi [26] is used.

The default settings in our experiments are as follows: each data center with a size 20, the number of data chunks $|K| = 10$, the task arrival rates $\lambda_k \in [0.01, 5], \forall k \in K$, the number of replicas $P = 3$, the data chunk size $\phi_k \in [0.01, 1], \forall k \in K$, and $D = 100$. We investigate how various parameters affect the overall computation, communication and overall cost by varying one parameter in each experiment group.

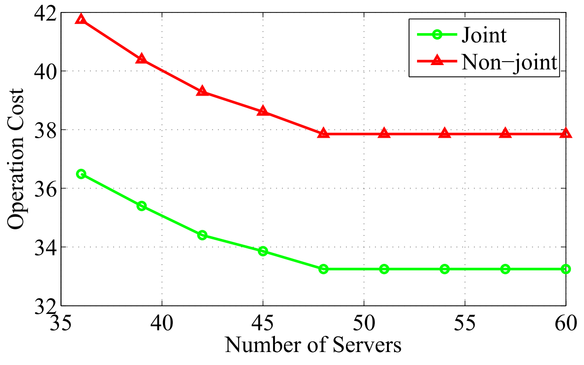
Fig. 3 shows the server cost, communication cost and overall cost under different total server numbers varying from 36 to 60. As shown in Fig. 3(a), we can see that the server cost always keep constant on any data center size. As observed from Fig. 3(b), when the total number of servers



(a)

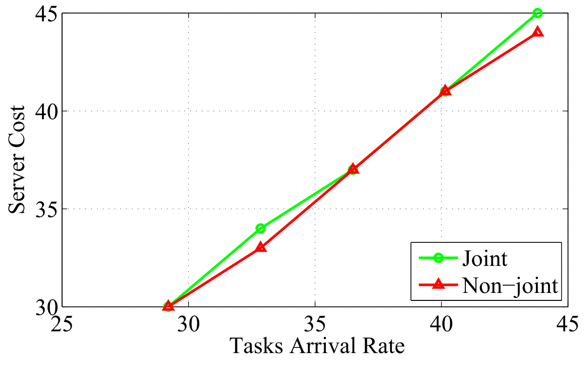


(b)

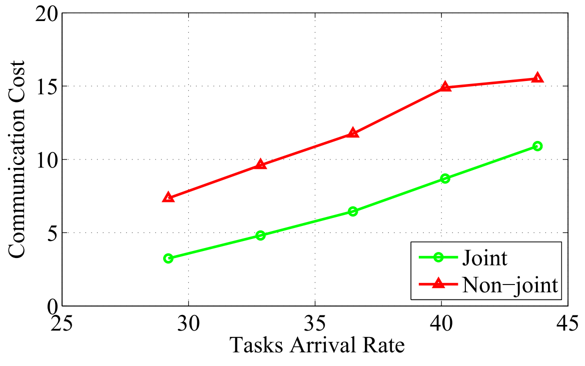


(c)

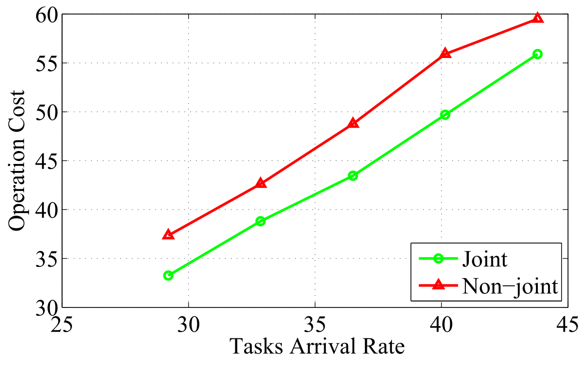
FIGURE 3. **On the effect of the number of servers. (a) Server Cost. (b) Communication Cost. (c) Overall Cost.**



(a)



(b)

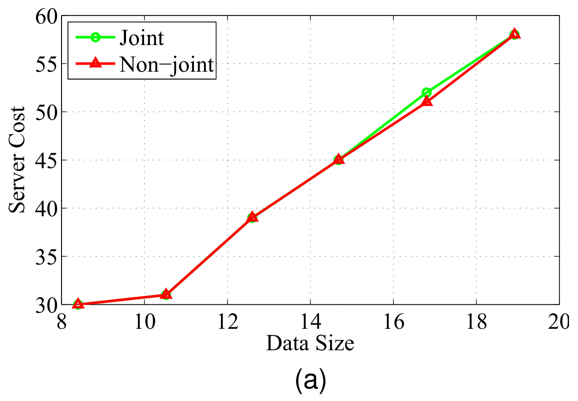


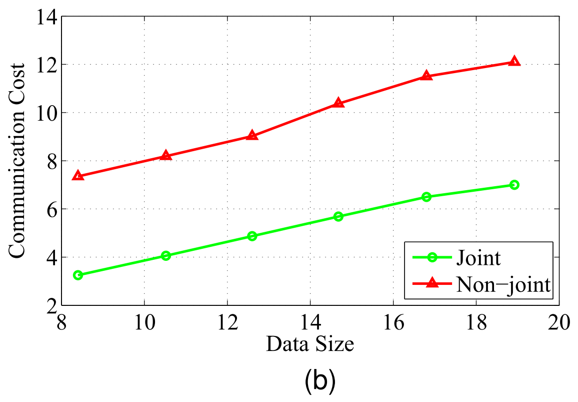
(c)

FIGURE 4. **On the effect of task arrival rate. (a) Server Cost. (b) Communication Cost. (c) Overall Cost.**

increases from 36 to 48, the communication costs of both algorithms decrease significantly. This is because more tasks and data chunks can be placed in the same data center when more servers are provided in each data center. Hence, the communication cost is greatly reduced. However, after the number of server reaching 48, the communication costs of both algorithms converge. The reason is that most tasks and their corresponding data chunks can be placed in the same data center, or even in the same server. Further increasing the number of servers will not affect the distributions of tasks or data chunks any more. Similar results are observed in Fig. 3(c).

Then, we investigate how the task arrival rate affects the cost via varying its value from 29.2 to 43.8. The evaluation results are shown in Fig. 4. We first notice that the total cost shows as an increasing function of the task arrival rates in both algorithms. This is because, to process more requests with the guaranteed QoS, more computation resources are needed. This leads to an increasing number of activated servers and hence higher server cost, as shown in Fig. 4(a). An interesting fact noticed from Fig. 4(a) is that "Joint" algorithm requires sometimes higher server cost than "Non-joint". This is because the first phase of the "Non-joint" algorithm greedily tries to lower the server cost. However, "Joint" algorithm balances the tradeoff between server cost and communication cost such that it incurs much lower communication cost and thus better results on the overall cost, compared to the "Non-joint" algorithm, as shown in Fig. 4(b) and (c), respectively.





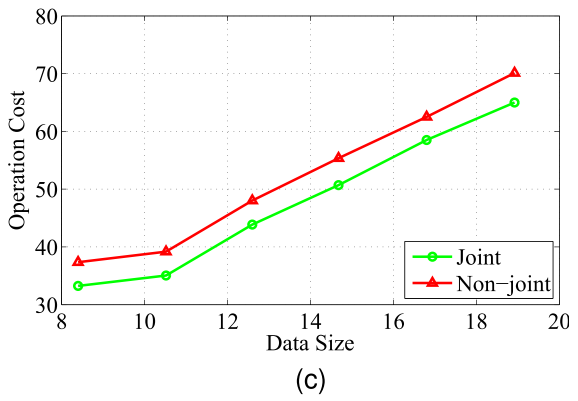
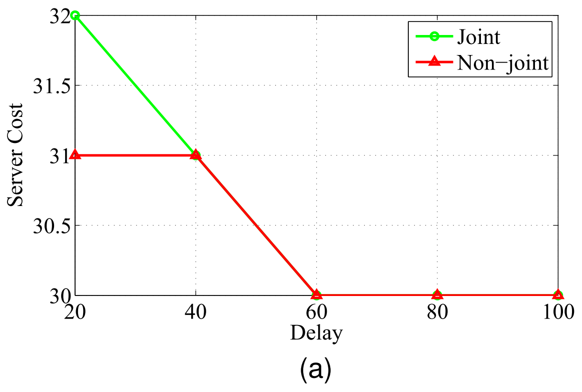
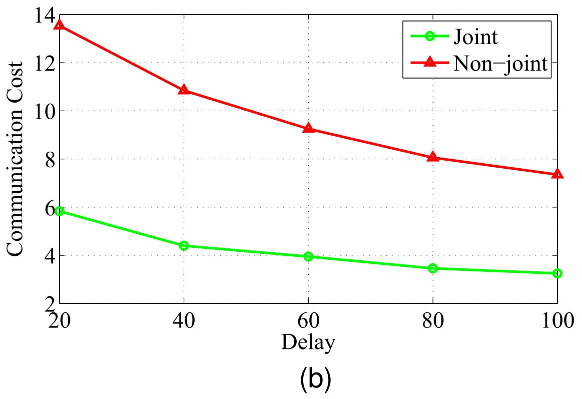


FIGURE 5. **On the effect of data size. (a) Server Cost. (b) Communication Cost. (c) Overall Cost.**





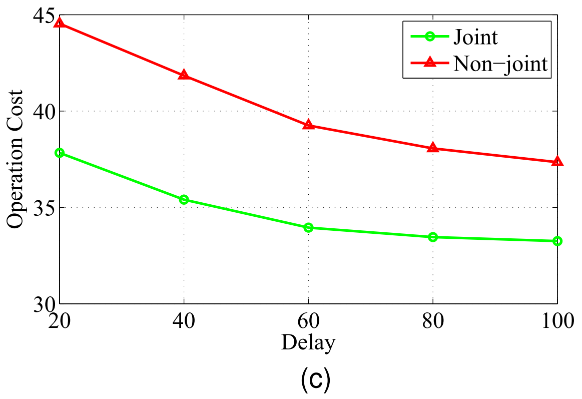


FIGURE 6. **On the effect of expected task completion delay. (a) Server Cost. (b) Communication Cost. (c) Overall Cost.**

Fig. 5 illustrates the cost as a function of the total data chunk size from 8.4 to 19. Larger chunk size leads to activating more servers with increased server cost as shown in Fig. 5(a). At the same time, more resulting traffic over the links creates higher communication cost as shown in Fig. 5(b). Finally, Fig. 5(c) illustrates the overall cost as an increasing function of the total data size and shows that our proposal outperforms ''Non-joint'' under all settings.

Next we show in Fig. 6 the results when the expected maximum response time D increases from 20 to 100. From Fig. 6(a), we can see that the server cost is a non-increasing function of D. The reason is that when the delay requirement is very small, more servers will be activated to guarantee the QoS. Therefore, the server costs of both algorithms decrease as the delay constraint increases. A looser QoS requirement also helps find cost-efficient routing strategies as illustrated in Fig. 6(b). Moreover, the advantage of our ''Joint'' over ''Non-joint'' can be always observed in Fig. 6(c).

Finally, Fig. 7 investigates the effect of the number of replicas for each data chunk, which is set from 1 to 6. An interesting observation from Fig. 7(c) is that the total cost first decreases and then increases with the increasing number of replicas. Initially, when the replica number increases from 1 to 4, a limited number of activated servers are always enough for task processing, as shown in Fig. 7(a). Meanwhile, it improves the possibility that task and its required data

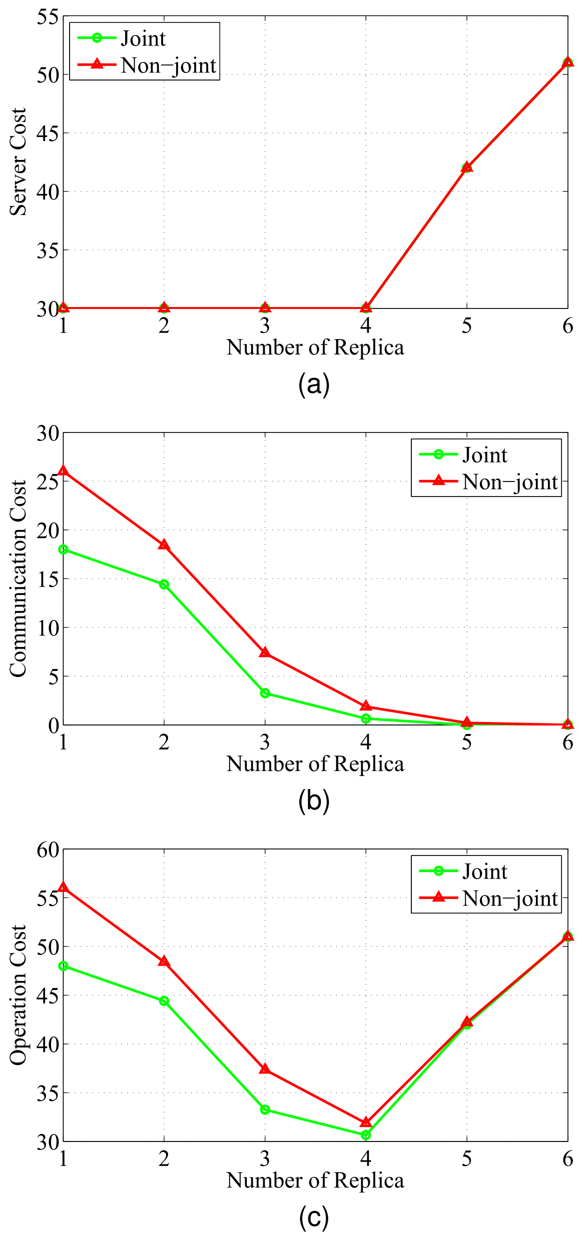


FIGURE 7. **On the effect of the number of replica. (a) Server Cost. (b) Communication Cost. (c) Overall Cost.**

chunk are placed on the same server. This will reduce the communication cost, as shown in Fig. 7(b). When the replica number becomes large, no further benefits to communication cost will be obtained while more servers must be activated only for the purpose of providing enough storage resources. In this case, the server and hence the overall costs shall be increased, as shown in Fig. 7(a) and (c), respectively.

Our discovery that the optimal number of chunk replicas is equal to 4 under the network setting above is verified one more time by solving the formulation given in Section 4.4 that is to minimize the number of replicas with the minimum total cost. Additional results are given under different settings via varying the task arrival rate and chunk size in the ranges of [0.1, λ_U] and [ϕ_L, 1.0], respectively, where a number of combinations of (λ_U, ϕ_L) are shown in Fig. 8. We observe that the optimal number of replica a non-decreasing function of the task arrival rate under the same chunk size while a non-increasing function of the data chunk size under the same task arrival rate.

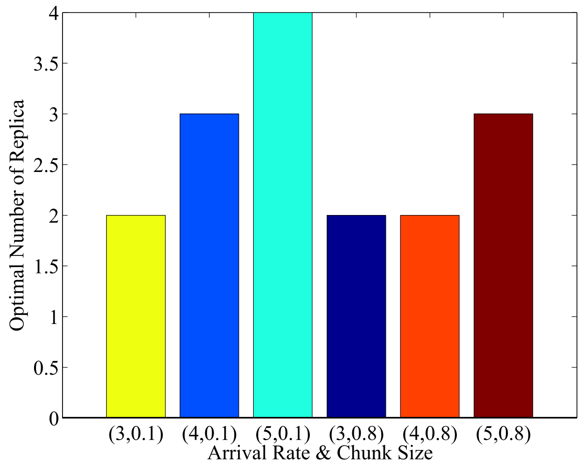


FIGURE 8. **Optimal number of replica.**

VII. CONCLUSION

In this paper, we jointly study the data placement, task assignment, data center resizing and routing to minimize the overall operational cost in large-scale geo-distributed data centers for big data applications. We first characterize the data processing process using a two-dimensional Markov chain and derive the expected completion time in closed-form, based on which the joint optimization is formulated as an MINLP problem. To tackle the high computational complexity of solving our MINLP, we linearize it into an MILP problem. Through extensive experiments, we show that our joint-optimization solution has substantial advantage over the approach by two-step separate optimization. Several interesting phenomena are also observed from the experimental results.

REFERENCES

[1] (2013). *Data Center Locations* [Online]. Available: http://www.google.com/about/datacenters/inside/locations/index.html

[2] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, and X. Zhu, “No ‘power’ struggles: Coordinated multi-level power management for the data center,” in *Proc. 13th Int. Conf. Archit. Support Program. Lang. Oper. Syst.*, 2008, pp. 48–59.

[3] L. Rao, X. Liu, L. Xie, and W. Liu, “Minimizing electricity cost: Optimization of distributed internet data centers in a multi-electricity-market environment,” in *Proc. 29th Int. Conf. Comput. Commun.*, 2010, pp. 1–9.

[4] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, “Greening geographical load balancing,” in *Proc. Int. Conf. Meas. Model. Comput. Syst.*, 2011, pp. 233–244.

[5] R. Urgaonkar, B. Urgaonkar, M. J. Neely, and A. Sivasubramaniam, “Optimal power cost management using stored energy in data centers,” in *Proc. Int. Conf. Meas. Model. Comput. Syst.*, 2011, pp. 221–232.

[6] H. Xu, C. Feng, and B. Li, “Temperature aware workload management in geo-distributed datacenters,” in *Proc. Int. Conf. Meas. Model. Comput. Syst.*, 2013, pp. 33–36.

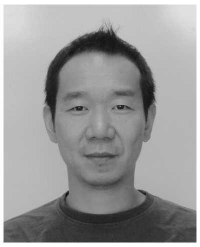
[7] J. Dean and S. Ghemawat, “Mapreduce: Simplified data processing on large clusters,” *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[8] S. A. Yazd, S. Venkatesan, and N. Mittal, "Boosting energy efficiency with mirrored data block replication policy and energy scheduler," *SIGOPS Oper. Syst. Rev.*, vol. 47, no. 2, pp. 33–40, 2013.
[9] I. Marshall and C. Roadknight, "Linking cache performance to user behaviour," *Comput. Netw. ISDN Syst.*, vol. 30, no. 223, pp. 2123–2130, 1998.
[10] H. Jin *et al.*, "Joint host-network optimization for energy-efficient data center networking," in *Proc. 27th Int. Symp. Parallel Distrib. Process.*, 2013, pp. 623–634.
[11] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, "Cutting the electric bill for internet-scale systems," in *Proc. ACM Special Interest Group Data Commun.*, 2009, pp. 123–134.
[12] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *Proc. 34th Annu. ISCA*, 2007, pp. 13–23.
[13] S. Govindan, A. Sivasubramaniam, and B. Urgaonkar, "Benefits and limitations of tapping into stored energy for datacenters," in *Proc. 38th Annu. ISCA*, 2011, pp. 341–352.
[14] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, "It's not easy being Green," in *Proc. ACM SIGCOMM*, 2012, pp. 211–222.
[15] Z. Liu *et al.*, "Renewable and cooling aware workload management for sustainable data centers," in *Proc. Int. Conf. Meas. Model. Comput. Syst.*, 2012, pp. 175–186.
[16] M. Sathiamoorthy *et al.*, "Xoring elephants: Novel erasure codes for big data," in *Proc. 39th Int. Conf. Very Large Data Bases*, 2013, pp. 325–336.
[17] B. Hu, N. Carvalho, L. Laera, and T. Matsutsuka, "Towards big linked data: A large-scale, distributed semantic data storage," in *Proc. 14th Int. Conf. Inf. Integr. Web Based Appl. Services*, 2012, pp. 167–176.
[18] J. Cohen, B. Dolan, M. Dunlap, J. M. Hellerstein, and C. Welton, "Mad skills: New analysis practices for big data," *Proc. VLDB Endowment*, vol. 2, no. 2, pp. 1481–1492, 2009.
[19] R. Kaushik and K. Nahrstedt, "T*: A data-centric cooling energy costs reduction approach for Big Data analytics cloud," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, 2012, pp. 1–11.
[20] F. Chen, M. Kodialam, and T. V. Lakshman, "Joint scheduling of processing and shuffle phases in mapreduce systems," in *Proc. 29th Int. Conf. Comput. Commun.*, 2012, pp. 1143–1151.
[21] H. Shachnai, G. Tamir, and T. Tamir, "Minimal cost reconfiguration of data placement in a storage area network," *Theoretical Comput. Sci.*, vol. 460, pp. 42–53, 2012.
[22] S. Agarwal, J. Dunagan, N. Jain, S. Saroiu, A. Wolman, and H. Bhogan, "Volley: Automated data placement for geo-distributed cloud services," in *Proc. 7th USENIX Symp. NSDI*, 2010, pp. 17–32.
[23] A. Cidon, R. Stutsman, S. Rumble, S. Katti, J. Ousterhout, and M. Rosenblum, "MinCopysets: Derandomizing replication in cloud storage," in *Proc. 10th USENIX Symp. NSDI*, 2013, pp. 1–5.
[24] S. Gunduz and M. Ozsu, "A poisson model for user accesses to web pages," in *Computer and Information Sciences-ISCIS* (Lecture Notes in Computer Science). Berlin, Germany: Springer-Verlag, 2003, pp. 332–339.
[25] L. Kleinrock, "The latency/bandwidth tradeoff in gigabit networks," *IEEE Commun. Mag.*, vol. 30, no. 4, pp. 36–40, Apr. 1992.
[26] (2009). *Gurobi* [Online]. Available: http://www.gurobi.com

LIN GU received the M.S. degree in computer science from the University of Aizu, Fukushima, Japan, in 2011, where she is currently pursuing the Ph.D. degree. She received the B.S. degree in computer science and technology from the School of Computer Science and Technology, Huazhong University of Science and Technology, China, in 2009. Her current research interests include cloud computing, big data, and software-defined networking.

DEZE ZENG received the Ph.D. and M.S. degrees in computer science from the University of Aizu, Aizu-Wakamatsu, Japan, in 2013 and 2009, respectively, where he is currently a Research Assistant. He received the B.S. degree from the School of Computer Science and Technology, Huazhong University of Science and Technology, China, in 2007. His current research interests include cloud computing, and networking protocol design and analysis, with a special emphasis on delay-tolerant networks and wireless sensor networks.

PENG LI received the B.S. degree from the Huazhong University of Science and Technology, China, the M.S. and Ph.D. degrees from the University of Aizu, Japan, in 2007, 2009, and 2012, respectively, where he is currently a Post-Doctoral Researcher. His research interests include networking modeling, cross-layer optimization, network coding, cooperative communications, cloud computing, smart grid, and performance evaluation of wireless and mobile networks for reliable, energy-efficient, and cost-effective communications.



SONG GUO (M'02–SM'11) received the Ph.D. degree in computer science from the University of Ottawa, Canada, in 2005. He is currently a Full Professor with the School of Computer Science and Engineering, University of Aizu, Japan. His research interests are mainly in protocol design and performance analysis for reliable, energy-efficient, and cost-effective communications in wireless networks. He received the Best Paper Awards at ACM Conference on Ubiquitous Information Management and Communication in 2014, the IEEE Conference on Computational Science and Engineering in 2011, and the IEEE Conference on High Performance Computing and Communications in 2008. He currently serves as an Associate Editor of the IEEE Transactions on Parallel and Distributed Systems. He is on the editorial boards of ACM/Springer Wireless Networks, Wireless Communications and Mobile Computing, the *International Journal of Distributed Sensor Networks*, the *International Journal of Communication Networks and Information Security*, and others. He has also been in organizing committee of many international conferences, including serving as a General Chair of MobiQuitous 2013. He is a Senior Member of the Association for Computing Machinery.