# Carbon-Aware Online Control of Geo-Distributed Cloud Services

Zhi Zhou, Fangming Liu, *Member, IEEE*, Ruolan Zou, Jiangchuan Liu, *Senior Member, IEEE*, Hong Xu, *Member, IEEE*, and Hai Jin, *Senior Member, IEEE*

**Abstract**—Recently, datacenter carbon emission has become an emerging concern for the cloud service providers. Previous works are limited on cutting down the power consumption of datacenters to defuse such a concern. In this paper, we show how the spatial and temporal variabilities of the electricity carbon footprint can be fully exploited to further green the cloud running on top of geographically distributed datacenters. Specifically, we first verify that electricity cost minimization conflicts with carbon emission minimization, based on an empirical study of several representative geo-distributed cloud services. We then jointly consider the electricity cost, service level agreement (SLA) requirement, and emission reduction budget. To navigate such a three-way tradeoff, we take advantage of Lyapunov optimization techniques to design and analyze a carbon-aware control framework, which makes online decisions on geographical load balancing, capacity right-sizing, and server speed scaling. Results from rigorous mathematical analysis and real-world trace-driven evaluation demonstrate the effectiveness of our framework in reducing both electricity cost and carbon emission.

**Index Terms**—Carbon reduction, load balancing, capacity right-sizing, geo-distributed datacenters, three-way tradeoff, online control

## 1 Introduction

Geographically distributed datacenters [1] that host cloud applications such as web search, social networks and video streaming have quickly ascended to the spotlight in terms of the enormous power demand and carbon emission. It is estimated that datacenters will consume about $8$ percent of the worldwide electricity by 2020, and produce $2.6$ percent of the global carbon emission [2]. As one of the leading cloud service providers, Google emitted $1.68 \times 10^6$ tons of carbon in 2011 [3], $15.86$ percent more than the emission of 2010, which is on par with the carbon emission of the United Nations headquarter [4].

Intuitively, carbon emission may be reduced by cutting down the energy consumption [5], [6], [7], [8], [9], [10], [11]. Existing approaches toward this direction fall into the following three categories at different spatial levels: *Geographical load balancing* at the geographic level, which utilizes the heterogeneity of energy/cooling efficiency of geo-distributed datacenters, and distributes more workload to datacenters with higher energy/cooling efficiency, thus to reduce the cooling power [11]. *Capacity right-sizing* at the datacenter level, which dynamically turns off redundant servers when demand decreases, thus to eliminate the power consumption of idle servers [8]. Finally, *server speed scaling* at the server level, which adjusts the CPU frequency based on the amount of workload served, thus to reduce the running power of the server [10].

While these pioneer works are effective in reducing energy consumption and cost, the savings do not necessarily translate into carbon emission reduction, particularly for geographically distributed datacenters. As illustrated in Fig. 1, the electricity carbon footprint exhibits strong spatial and temporal variability: different regions generate electricity with their respective fuel mixes, and have different carbon footprints. The time-varying fuel mix also leads to temporal differences in carbon footprint even for the same location. Therefore, by routing more requests to Alberta that has colder weather and thus higher cooling efficiency, the total energy consumption can be reduced. However, the total carbon emission would increase since Alberta has a high carbon emission rate. Our empirical study in the next section further demonstrates that greener energy is generally more expensive, and hence there is a tradeoff between electricity cost and carbon emission minimization. Specifically, for real world geo-distributed cloud platforms, e.g., Google, Amazon and Microsoft, electricity cost minimization often conflicts with carbon emission minimization.

The notion of *carbon tax* has been explored for accommodating such variability and diversity, and it has been found that $10$ percent carbon emission reduction can be achieved without extra cost [2]. Unfortunately, current carbon tax remains low to effectively motivate providers to reduce carbon output, and the carbon tax itself is only one of the three key policies to inhibit carbon emission. The other two policies, "Cap and Trade" and "Baseline and Credit" that enforce a carbon source to operate within an allowance or budget [12], are yet to be explored. Moreover, carbon neutrality (i.e., reducing the net carbon footprint to zero) has


- *Z. Zhou, F. Liu, R. Zou and H. Jin are with the Services Computing Technology and System Lab, Cluster and Grid Computing Lab in the School of Computer Science and Technology, Huazhong University of Science and Technology, 1037 Luoyu Road, Wuhan 430074, China. E-mail: {zhiz, fmliu, hjin}@mail.hust.edu.cn.*
- *J. Liu is with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada. E-mail: jcliu@cs.sfu.ca.*
- *H. Xu is with the Department of Computer Science, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong, China. His work is partly supported by HKRGC-ECS 21201714, CityU HK Start-up grant 7200366. E-mail: henry.xu@cityu.edu.hk.*


*Manuscript received 11 Feb. 2015; revised 28 Oct. 2015; accepted 15 Nov. 2015. Date of publication 3 Dec. 2015; date of current version 10 Aug. 2016. Recommended for acceptance by R. Kwok.*
*For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.*
*Digital Object Identifier no. 10.1109/TPDS.2015.2504978*

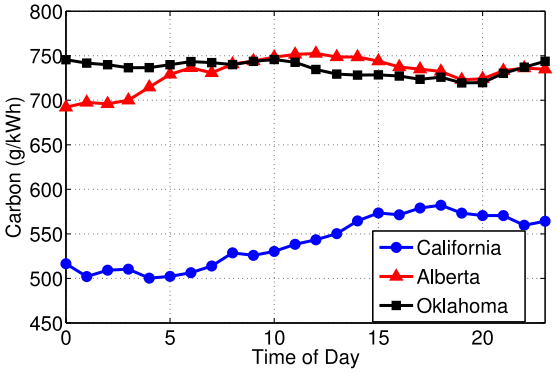



Fig. 1. Carbon emission per kWh electricity at three different locations in north America on September 30th 2012. Data is provided by each Regional Transmission Organization (RTO) [12].

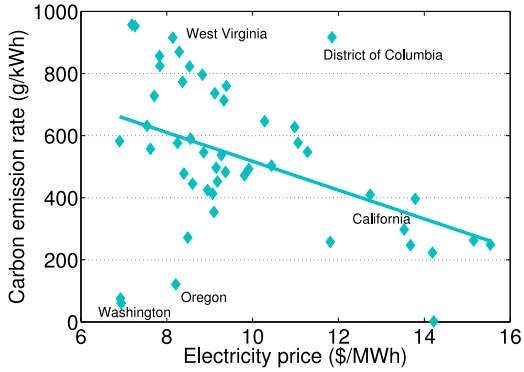



Fig. 2. Carbon emission versus electricity price in the continental US We can observe a clear negative correlation between the two.

been increasingly set as a strategic goal by companies such as Google, Microsoft and Facebook, which is generally achieved by offsetting the carbon footprint with the purchased renewable energy certificates (RECs). Given a budget on RECs, the carbon emission needs to be capped. The challenge toward this target however is enormous: given limited prior knowledge of the bursty workload, how can a cloud operator make dynamic decisions to minimize the energy cost, while maintaining the long-term emission budget? Since the future information is hard to be accurately predicted in practice, an online algorithm that can make dynamic decisions based on the historical and current information is needed.

In this paper, we provide strong evidence that the electricity cost, service level agreement (SLA) requirement, and emission reduction budget must be jointly considered towards reducing carbon emission for geographically distributed datacenters. We present a coherent carbon-aware online control framework that navigates such a three-way tradeoff. We rigorously design and analyze the control mechanism using Lyapunov optimization [13], [14], which effectively incorporates the long-term carbon emission constraints into real-time optimization. Our framework dynamically makes decisions across different levels for geographical load balancing, capacity right-sizing, and server speed scaling. Specifically, in the service level, we determine how to distribute user requests to appropriate datacenters according to the current electricity prices and carbon emission rates; in the datacenter level, we determine how many servers to activate at each datacenter; and in the server level, we determine how to set the service rate of each activated server.

Our framework is easily tunable by a control parameter $V$ that represents the relative importance of cost minimization versus emission enforcement, and facilitates a provable $[O(1/V), O(V)]$ cost-emission tradeoff. With such a tradeoff, the geo-distributed cloud can achieve a time-averaged electricity cost arbitrarily close to the optimum, while still maintaining the long-term carbon emission budget. Through an empirical evaluation using the real-world electricity generation and price data and workload traces from Microsoft's enterprise storage systems [15], we show that our solution is practical to achieve a specified long-term emission reduction target, without incurring excessive cost.

The rest of this paper is organized as follows. Different from and complementary to our preliminary work [16], we conduct an empirical study on the cost-emission tradeoff in realistic geo-distributed cloud services in Section 2. In Section 3, we improve the three-way tradeoff model by replacing the queuing delay in our preliminary work [16] with the wide-area network latency, since the latter dominates user-perceived latency in practice. We further develop a new carbon-aware online control framework and demonstrate its effectiveness in Section 4. Compared to our preliminary work [17], convex optimization technique is newly introduced to address the more complicated optimization model. The framework is evaluated in Section 5 with more comprehensive trace-driven simulations than our preliminary work [16]. Finally, Section 6 surveys the related work and Section 7 concludes the paper.

## 2 Empirical Study of Geo-Distributed Cloud Services

To motivate the joint optimization on energy cost and carbon emission, we take empirical study to demonstrate the tradeoff between energy cost and carbon emission for geo-distributed cloud services. We begin our study with an empirical analysis of the annual electricity generation and price data from 48 states in the continental US, (including the District of Columbia). In Fig. 2, we plot the carbon emission rates and the electricity prices according to data in *Electric Power Annual 2012* from the US Energy Information Administration's website [17], [18]. It clearly shows that there is a negative correlation between carbon emission and electricity price (correlation coefficient $-0.43$), implying the existence of a long-term cost-emission tradeoff at the national level. In particular, the electricity in regions such as West Virginia is cheap but dirty (as it is depicted in the upper-left quarter of Fig. 2), while regions such as California have expensive but clean energy. There are a few exceptions however: states like Oregon and Washington enjoy both cheap and clean electricity; yet the District of Columbia has both expensive and dirty electricity. Intuitively, Oregon and Washington are preferred locations to host datacenters when considering electricity and environmental costs.

We now look into datacenters from five representative cloud service providers, including Google [19], Microsoft [20], Amazon [21], Facebook [22], and Apple [23]. In Fig. 3, we depict their datacenter locations in the US All of them have geo-distributed datacenters; Google deploys the largest number of datacenters (5), whereas Amazon, which has the

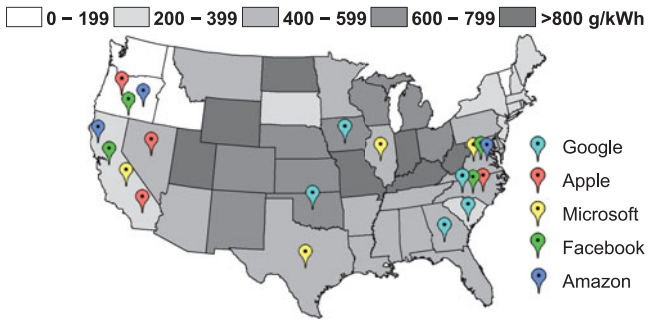



Fig. 3. Carbon emission rate of each state and the datacenter locations of five clouds in the US Darker colors imply higher carbon footprint of electricity generation.

least number, deploys 3 still (not including GovCloud). Half of the 20 datacenters are located in "clean" regions with carbon footprint lower than 400 g/kWh. 18 datacenters are located in regions with carbon footprint lower than 600 g/kWh, and no datacenter is located in "dirty" regions with carbon footprint higher than 800 g/kWh. Overall, the datacenter locations of the five cloud services are highly concentrated in states such as California, Oregon, North Carolina, and Virginia, which either enjoy very clean electricity as we have shown, or have world-class network infrastructures.

To further understand the existence of the cost-emission tradeoff in the geo-distributed cloud services, we plot the yearly carbon emission rate versus the electricity price of different clouds' datacenters in Fig. 4, together with their correlation coefficients. The results show that there are strong negative correlations between the annual carbon emission rate and the electricity price for Google and Microsoft datacenters. The correlation coefficients are $-0.842$ and $-0.84$, respectively, much higher than the national level ($-0.43$).

On the other hand, the same tradeoff is not clearly observable for Amazon, Facebook and Apple datacenters. A further examination shows that these three providers each builds a datacenter in Oregen where the electricity is both cleanest and cheapest, thus offsetting the negative correlations exhibited by the remaining datacenters. Excluding Oregon and focusing on the remaining datacenters in Fig. 5, we can observe a strong negative correlation between carbon emission and electricity price. This observation

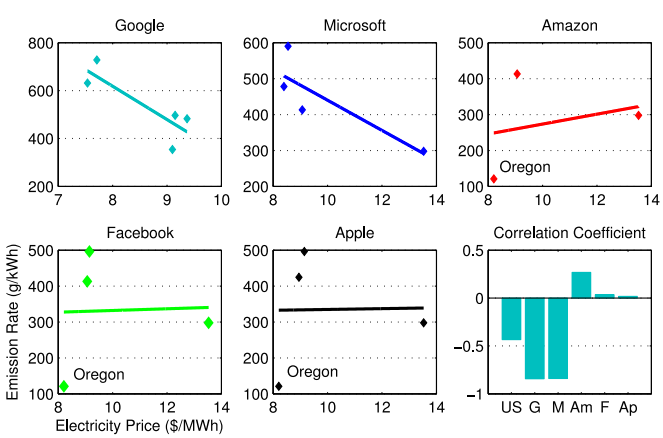



Fig. 4. Carbon emission rate versus electricity price in states hosting geo-distributed datacenters of Google, Microsoft, Amazon, Facebook and Apple.

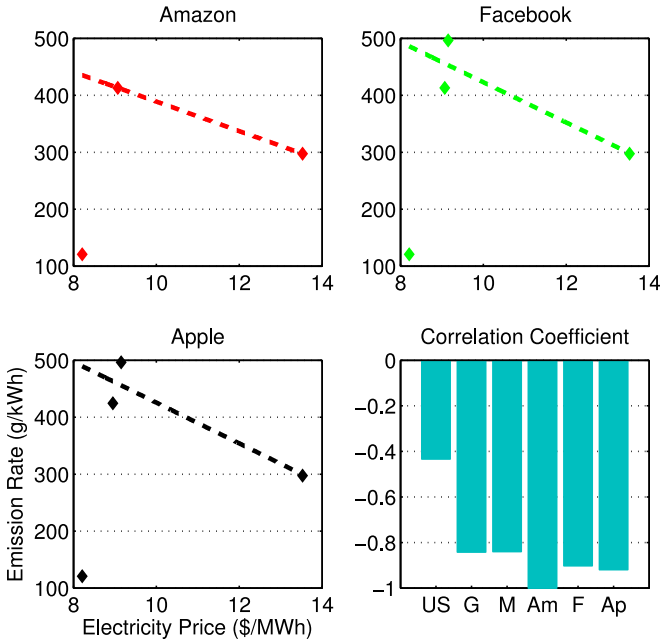



Fig. 5. Carbon emission rate versus electricity price in states hosting geo-distributed datacenters of Amazon, Facebook and Apple (excluding Oregon).

indicates that, after routing much traffic to Oregan to minimize both electricity cost and carbon emission, routing the remaining traffic to the remaining datacenters would also induce a cost-emission tradeoff.

To summarize, we have quantitatively verified the existence of long-term cost-emission tradeoff in realistic geo-distributed cloud services. This strongly suggests that, for geo-distributed cloud services, it needs a joint optimization on energy cost and carbon emission, rather than the naive cost reduction approach which can deteriorate the carbon emission. We are now ready to propose the carbon-aware online control framework for geo-distributed cloud services, which jointly optimizes energy cost and carbon emission, to achieve a delicate $[O(1/V), O(V)]$ cost-emission tradeoff.

## 3 The Three-Way Tradeoff Model

We consider a provider running cloud services with $N$ geo-distributed datacenters, denoted by $\mathcal{D} = \{1, 2, \ldots, N\}$. Each datacenter $j \in \mathcal{D}$ deploys $M_j$ servers. For ease of exposition, we assume that all the servers in one datacenter are homogeneous. The cloud deploys $M$ front-end proxy servers, denoted by $\mathcal{S} = \{1, 2, \ldots, M\}$ in various regions to direct user requests to the appropriate datacenters. Inspired by [5], we consider a discrete time model. In every time slot $t = (0, 1, 2, \ldots, \tau, \ldots)$, user requests arrive and aggregate at each front-end proxy server. We use $A_i(t), \forall i \in \mathcal{S}$ to denote the request arrival rate at front-end proxy server $i$ during time slot $t$. In practice, the time slot length matches the time scale at which the electricity price or carbon emission rate is updated. A typical setting is 10 minutes (a value that well exploits the temporal diversity of both electricity price and carbon footprint, without incurring excessive overhead of switching the servers on/off frequently), which is also used in our trace-driven performance evaluation. Table 1 summarizes the key notations used throughout this paper.

TABLE 1
Key Parameters in our Model

| Notations | Definitions |
|---|---|
| $M$ | The number of front-end proxy servers $\mathcal{S}$ |
| $N$ | The number of geographically distributed datacenters $\mathcal{D}$ |
| $A_i(t)$ | The request arrival rate at front-end proxy server $i$ during time slot $t$ |
| $R_{ij}(t)$ | The amount of request routing from the front-end server $i$ to the datacenter $j$ |
| $m_j(t)$ | The number of servers to be activated in datacenter $j$ at time slot $t$ |
| $\mu_j(t)$ | The service rate of each active server in datacenter $j$ at time slot $t$ |
| $E_j(t)$ | The power consumption of datacenter $j$ at time slot $t$ |
| $L_{ij}$ | The round-trip network latency between the front-end server $i$ and datacenter $j$ |
| $C_j(t)$ | The carbon emission rate in location $j$ at time slot $t$ |
| $C_j$ | The long-term time-averaged carbon emission budget for datacenter $j$ |
| $P_j(t)$ | The electricity price in location $j$ at time slot $t$ |
| $Q_j(t)$ | The backlog of the virtual queue for datacenter $j$ at time slot $t$ |
| $V$ | Lyapunov control parameter |

### 3.1 Control Decisions

Our online control framework has three levels of control decisions.

#### 3.1.1 Geographical Load Balancing

In the geographical level, the energy cost and the carbon emission can be reduced by balancing the workload across the datacenters, according to the spatial variability of energy price and carbon output. In each time slot $t$, given the request arrival rate $A_i(t)$ at the proxy server $i$, the control decision is to update the request routing from the proxy $i$ to datacenter $j$, denoted as $R_{ij}(t), \forall i \in \mathcal{S}, \forall j \in \mathcal{D}$. Therefore, we have

$$\sum_{j=1}^{N} R_{ij}(t) = A_i(t), \quad \forall i \in \mathcal{S}, \tag{1}$$

$$R_{ij}(t) \geq 0. \tag{2}$$

#### 3.1.2 Datacenter Right Sizing

In the datacenter level, energy cost and carbon emission can be reduced by dynamically adjusting the number of active servers, known as "right sizing" [5]. Let $m_j(t)$ denotes the number of servers to be activated in datacenter $j$ in time slot $t$. Since a cloud datacenter typically contains thousands of active servers, the integer constraints on $m_j(t), \forall j \in \mathcal{D}, \forall t$ can be relaxed and we can treat $m_j(t)$ as a continuous variable. Therefore, $m_j(t)$ should satisfy the following constraint:

$$0 \leq m_j(t) \leq M_j, \quad \forall j \in \mathcal{D}. \tag{3}$$

#### 3.1.3 Server Speed Scaling

In the server level, energy cost and carbon emission can be reduced by adjusting the CPU frequency, known as "speed scaling" [10]. Let $\mu_j(t)$ denotes the service rate of each active server in datacenter $j$. Typically, for each $\mu_j(t)$, it cannot exceed the maximum service rate $s_j$, which follows

$$0 \leq \mu_j(t) \leq s_j, \quad \forall j \in \mathcal{D}. \tag{4}$$

Besides, to ensure that the workload routed to each datacenter could be completely served, the following *datacenter capacity constraint* (5) should be enforced, i.e., the amount of workload $\sum_i R_{ij}(t)$ assigned to each datacenter $j$ cannot exceed the later's total service capacity $m_j u_j$:

$$\sum_{i=1}^{M} R_{ij}(t) \leq m_j \mu_j, \quad \forall j \in \mathcal{D}. \tag{5}$$

### 3.2 Power Consumption Model

It has been demonstrated that [7], the amount of power consumed by a server running at speed $\mu$ can be characterized by $\alpha\mu^{\nu} + \beta$, where $\alpha$ is a positive factor, $\beta$ represents the power consumption in the idle state, and the exponent parameter $\nu$ is empirically determined as $\nu \geq 1$, with a typical value of $\nu = 2$ in practice [7].

Given the number of active servers $m_j(t)$, parameters $\alpha_j, \beta_j, \nu_j$, and the power usage efficiency metric $\mathrm{PUE}_j$ in datacenter $j$, the power consumption of datacenter $j$ in time slot $t$ can then be quantified by $E_j(t)$ as follows:

$$E_j(t) = \mathrm{PUE}_j \cdot m_j(t) \cdot [\alpha_j \mu_j^{\nu_j}(t) + \beta_j], \tag{6}$$

PUE is defined as the ratio of the total amount of power used by the entire facility to the power delivered to the computing equipment [24].

Though many datacenters are investing renewable power to power themselves, here we assume that each datacenter is completely powered by the power grid. Note that this is realistic, specifically, internet giants such as Google [25], Microsoft [26] and Facebook [27] are both investing large-scale wind or solar farms to "clean up" their cloud service. However, the carbon reduction is realized by signing power purchase agreements (PPAs) [25] with renewable energy operators, yet the renewable power is not really transmitted to the datacenters. Instead, the renewable power is sold back to the power grid, and the datacenters still completely rely on the power gird. By applying those renewable energy credits (RECs) [25] realised from the renewable generation, the carbon emission of the consumed grid power can be partially or fully offset.

### 3.3 Latency SLA Model

For interactive applications as web search and social networking services, latency is the most critical performance metric [28]. In this paper, we focus on the end-to-end request latency from a front-end proxy server to a processing datacenter in wide-area network, as it largely accounts for the user-perceived latency and overweighs other factors such as queuing or processing delays at datacenters [29]. We assume that the geo-distributed datacenters are connected by a private backbone network [30]. Specifically, the round-trip times within large datacenters with tens of thousands of servers are typically $200{-}500$ $\mu$s, while with the long-term advances in operation system and hardware, datacenter round-trips low to $1\mu$s can be also achievable [31]. However,

TABLE 2
Carbon Dioxide Emission Per Kilowatt-Hour for the Most Common Fuel Types [2]

| Fuel Type | Nuclear | Coal | Gas | Oil | Hydro | Wind |
|---|---|---|---|---|---|---|
| $CO_2$ g/kWh | 15 | 968 | 440 | 890 | 13.5 | 22.5 |

in a sharp contrast, the wide-area network latency that involves propagation, queuing, transmission, and nodal processing times in geo-distributed system is far longer, typically tens or hundreds of milliseconds. For example, a typical round-trip time between datacenters from California to Texas is about 31.8 ms, and that from California to New Jersey is about 73.5 ms [32].

The round-trip network latency $L_{ij}$ between the front-end server $i$ and datacenter $j$ can be obtained through active measurements or other means in practice [33]. Empirical studies have also demonstrated that, in backbone networks, the round-trip network latency $L_{ij}$ can be approximated by geographical distance $d_{ij}$ between the front-end server $i$ and datacenter $j$ as: $L_{ij} = d_{ij} \times 0.02\,\mathrm{ms/km}$ [30].

The average network latency of the front-end proxy server $i$ is then $L_i = \sum_{j\in\mathcal{D}} R_{ij}(t)L_{ij}/A_i$. To provide satisfiable experience to users, we enforce the following constraint that $L_i \le L_i^{\max}$, where $L_i^{\max}$ is the maximal tolerable response delay at the front-end server $i$. Therefore, we have the following SLA constraint:

$$\frac{\sum_{j\in\mathcal{D}} R_{ij}(t)L_{ij}}{A_i} \le L_i^{\max}. \tag{7}$$

### 3.4 Long-Term Carbon Reduction Model

To characterize the spatial and temporal variability of the carbon emission rate, we use the electricity generation data from each Regional Transmission Organization (RTO)'s website. We retrieve the real-time electricity fuel mix of all states for the seven major types of fuel (e.g., the real-time data of New England is updated every 5 minutes in [34]). Summing up the weighted contribution from each fuel type, we can estimate the carbon emission rate in location $j$ at time slot $t$ as follows

$$C_j(t) = \frac{\sum e_{kj}(t) \times c_k}{\sum e_{kj}(t)}, \tag{8}$$

where $e_{kj}(t)$ represents the electricity generated from fuel type $k$ in location $j$ at time slot $t$, and $c_k$ (measured in $g/kWh$) is the carbon emission rate of fuel type $k$ given in Table 2.

Given $C_j(t)$ and power consumption $E_j(t)$ in Eq. (6), the corresponding carbon emission becomes $E_j(t)\cdot C_j(t)$. In practice, most datacenters are operated within a certain carbon emission budget in a given *long time* interval (usually one year or longer). We therefore impose a long-term time-averaged carbon emission budget $C_j$ for each datacenter $j$ to reduce the carbon emission

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\{E_j(t)\cdot C_j(t)\} \le C_j. \tag{9}$$

### 3.5 Characterizing the Three-Way Tradeoff

With the SLA constraint and the long-term carbon reduction constraint, our objective is to minimize the long-term electricity cost. Specifically, given the power consumption $E_j(t)$ and electricity price $P_j(t)$ in datacenter $j$, the total electricity cost of $N$ datacenters at time slot $t$ can be quantified by $\sum_{j=1}^{N} E_j(t)P_j(t)$. The optimization of the three-way tradeoff, which jointly considers the electricity cost, the SLA requirement, and the carbon emission reduction under the control decisions $R_{ij}(t), m_j(t)$ and $\mu_j(t)$, can then be formulated as the following stochastic program

$$\min \limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1}\sum_{j=1}^{N} \mathbb{E}\{E_j(t)P_j(t)\}, \tag{10}$$

subject to (1), (2), (3), (4), (5), (7), and (9).

**Remark 1.** The above three-way tradeoff model is more amenable to practical implementation when compared to the models in [2]. These earlier works transform both the carbon emission and the workload latency to monetary cost, and arbitrate the three-way tradeoff by minimizing the aggregated cost. It is however nontrivial to precisely map the carbon emission, especially the workload latency to economic cost. In our model, capping the long-term carbon emission and real-time workload latency is more practical and accessible in realistic cloud service.

Since the datacenters' workload as well as the carbon emission rate is time-varying and unpredictable, the challenge of solving problem (10) is that, how can we guarantee the current control decisions are able to minimize the time-averaged electricity cost, while still maintaining the long-term carbon emission budget?

## 4 Carbon-Aware Online Control Framework

To solve stochastic programming problems involving uncertainly, various approaches have been proposed in literature. For example, by assuming a perfect knowledge of the near-future information within a look-ahead window, the randomized fixed horizon control (RFHC) [35] method can approximately decompose the long-term optimization into a series of deterministic optimization problems. Besides, given the statistical distribution of the uncertain information, genetic algorithm [36] can be applied to maximize the expectation of the objective. Unfortunately, in a cloud computing environment with bursty workload, neither the near-further information nor the statistical distribution of the workload can be accurately obtained.

The challenge of the optimization problem (10) is mainly posed by the time-coupling carbon emission constraint (9). Intuitively, from the view of queuing theory, the constraint (9) can be interpreted as the queue stability control, i.e., the time-averaged arrival $\limsup_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\{E_j(t)\cdot C_j(t)\}$ cannot exceed the service rate $C_j$. Fortunately, the queue stability problem has been well studied in queue theory [14], and Lyapunov optimization is a powerful technique to maintain the stability in an online manner, without requiring any future knowledge or statistical distribution of the uncertain information. With this insight, we design a Carbon-aware

Online Control Algorithm (COCA), which explicitly transforms the long-term objective and constraints to a series of real-time optimizations at each time slot. COCA is provably-efficient to achieve time-averaged electricity cost arbitrarily close to the optimum, while still maintaining the long-term carbon emission budget.

### 4.1 Problem Transformation Using Lyapunov Optimization

We first transform the long-term carbon emission constraint (9) into a well-studied queue stability problem. To this end, we introduce *virtual queues* $Q_j(t)$ for each datacenter $j$. Initially, we define $Q_j(0) = 0, \forall j \in \mathcal{D}$, and then update the queues per each time slot as follows

$$Q_j(t+1) = \max[Q_j(t) - C_j + E_j(t)C_j(t),\ 0], \tag{11}$$

where $C_j$, $E_j(t)$, and $C_j(t)$ are defined in Section 3.4. $\forall j \in \mathcal{D}$, $E_j(t)C_j(t)$ can be viewed as the "arrivals" of virtual queue $Q_j(t)$, and the constant $C_j$ can be viewed as the service rate of such a virtual queue.

Intuitively, the value of $Q_j(t)$ is the historical measurement of the backlog between the time-averaged emission during the interval $[0, t-1]$ and the emission budget $C_j$. A large value of $Q_j(t)$ implies that the emission during the interval $[0, t-1]$ exceeds the budget $C_j$. In fact, the carbon emission constraint (9) for each datacenter is enforced on the condition that the virtual queue $Q_j(t)$ is stable, i.e., $\lim_{T\to\infty} \mathbb{E}\{Q_j(T)\}/T = 0$. Specifically, from Eq. (11) it is clear that $Q_j(t+1) \geq [Q_j(t) - C_j + E_j(t)C_j(t)]$. Given this inequality over time slots $t \in \{0, 1, \ldots, T-1\}$ and then dividing the result by $T$, we have

$$\frac{Q_i(T) - Q_j(0)}{T} + C_j \geq \frac{1}{T}\sum_{t=0}^{T-1} E_j(t)C_j(t).$$

With $Q_j(0) = 0$, taking expectations of both sides yields

$$\lim_{T\to\infty} \frac{\mathbb{E}\{Q_j(t)\}}{T} + C_j \geq \lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\{E_j(t)C_j(t)\}. \tag{12}$$

If the virtual queues $Q_j(T)$ are stable, then $\lim_{T\to\infty} \mathbb{E}\{Q_j(T)\}/T = 0$ (the proof of the strong stability of virtual queues $Q_j(T)$ can be found in Theorem 1 later). Subtracting this into (12) yields the inequality (9).

#### 4.1.1 Characterizing the Emission-Cost Tradeoff

Let $\mathbf{Q}(t) = (Q_j(t))$ denotes the vector of all the virtual queues. We define the Lyapunov function as follows

$$L(\mathbf{Q}(t)) = \frac{1}{2}\sum_{j=1}^{N} Q_j^2(t). \tag{13}$$

This represents a scalar metric of the congestion level [14] in all virtual queues. For example, a small value of $L(\mathbf{Q}(t))$ implies that all the queue backlogs are small. The implication is that all the virtual queues have *strong stability*.

To keep the virtual queues stable (i.e., to enforce the emission budget) by persistently pushing the Lyapunov function towards a lower congestion state, we introduce $\Delta(\mathbf{Q}(t))$ as the *one-step conditional Lyapunov drift* [14]:

$$\Delta(\mathbf{Q}(t)) = \mathbb{E}\{L(\mathbf{Q}(t+1)) - L(\mathbf{Q}(t))|\mathbf{Q}(t)\}.$$

Under the Lyapunov optimization, the underlying objective of our optimal control decisions $R_{ij}(t), m_j(t)$ and $\mu_j(t), \forall i \in \mathcal{S}, \forall j \in \mathcal{D}, \forall t$ is to minimize a supremum bound on the following *drift-plus-cost* expression in each time slot:

$$\Delta(\mathbf{Q}(t)) + V\mathbb{E}\left\{\sum_{j=1}^{N} E_j(t)P_j(t)\right\}. \tag{14}$$

**Remark 2.** The control parameter $V(\geq 0)$ represents *a design knob* of the emission-cost tradeoff, i.e., how much we shall emphasize the cost minimization (Problem (10)) compared to emission budget (Constraint (9)). It empowers datacenter operators to make flexible design choices among the various tradeoffs between the carbon emission and the electricity cost. For example, one may prefer to incur an expected cost as small as possible, while keeping $\Delta(\mathbf{Q}(t))$ small to avoid exceeding the carbon emission budget.

#### 4.1.2 Bounding Drift-Plus-Cost

To derive the supremum bound of the *drift-plus-cost* expression given in Eq. (14), we need the following lemma.

**Lemma 1.** *In each time slot $t$, given any possible control decisions $m_j(t), \mu_j(t)$ and $R_{ij}(t)$, the Lyapunov drift $\Delta(\mathbf{Q}(t))$ can be deterministically bounded as follows:*

$$\Delta(\mathbf{Q}(t)) \leq B - \sum_{j=1}^{N} Q_j(t)\mathbb{E}\{C_j - E_j(t)C_j(t)|\mathbf{Q}(t)\}, \tag{15}$$

*where the constant $B \triangleq \frac{1}{2}(\sum_{j=1}^{N} C_j^2 + NC_{max}^2)$, and $C_{max} = \max_{j,t}\{E_j(t)C_j(t)\}$.*

The proof of this lemma can be found in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2015.2504978.

Based on Lemma 1, adding expression $V\mathbb{E}\{\sum_{j=1}^{N} E_j(t)P_j(t)|\mathbf{Q}(t)\}$ to both sides of Eq. (15) yields an upper bound of *drift-plus-cost* expression of the datacenter system

$$\begin{aligned}\Delta(\mathbf{Q}(t)) + V\mathbb{E}\left\{\sum_{j=1}^{N} E_j(t)P_j(t)|\mathbf{Q}(t)\right\} &\leq B - \sum_{j=1}^{N} Q_j(t)C_j \\ &+ \sum_{j=1}^{N} \mathbb{E}\{E_j(t)[VP_j(t) + Q_j(t)C_j(t)]|\mathbf{Q}(t)\}.\end{aligned} \tag{16}$$

### 4.2 Carbon-Aware Online Control Algorithm

Directly minimizing the drift-plus-cost expression in Eq. (14) involves implicit $\max[*]$ terms in Eq. (11). Without undermining the optimality, we seek to minimize the supremum bound, which is equivalent to maximizing the right side of inequality (16)), as in [14].

The long-term optimization (10) is now transformed to the following optimization at each time slot $t$

$$\min \sum_{j=1}^{N} E_j(t)[VP_j(t) + Q_j(t)C_j(t)], \quad \text{s.t.}(1),(2),(3),(4),(5),(7) \text{ are satisfied.} \tag{17}$$

**Algorithm 1.** Carbon-aware Online Control Algorithm (COCA)

**1)** In the beginning of each time slot $t$, observe the current queue backlog $Q_j(t)$ and other information $P_j(t)$ and $C_j(t)$ at each datacenter $j$.

**2)** Determine the control decisions $m_j(t), \mu_j(t)$ and $R_{ij}(t)$, $\forall i \in \mathcal{S}, \forall j \in \mathcal{D}$ to minimize the term $\sum_{j=1}^{N} \mathbb{E}\{E_j(t)[VP_j(t) + Q_j(t)C_j(t)]|\mathbf{Q}(t)\}$ in the right-hand-side of inequality (16).

**3)** Update the queues $\mathbf{Q}(t+1)$ according to Eq. (11) and the newly determined control decisions.

**Remark 3.** The transformed problem (17) embodies an economic interpretation. At each time slot $t$, it strives to minimize the total cost of current power consumption and the penalty of carbon emission, as priced by the queue backlog $\mathbf{Q}(t)$. This balances our interest in minimizing the long-term electricity cost and enforcing the long-term carbon emission within the predefined budget, and $V$ is the control knob to adjust our emphasis on cost minimizing compared to emission enforcement.

In fact, the transformed problem that minimizes the supremum bound of the *drift-plus-cost* may still be complicated. As in our case, the problem (17) is nonlinear and nonconvex, mixed with variables that can not be easily decomposed. Then, a natural question to ask is, what happens when COCA does not *accurately* minimize the transformed Problem (17). Excitingly, we find that COCA is *robust* against the minimization errors of the problem (17). Specifically, the following theorem shows the optimality of the COCA algorithm under a minimization error $\delta$, in terms of a tradeoff between the cost minimization and emission enforcement. The minimization error $\delta$ of the Problem (17) means that, $\forall t, \widehat{S}(t) - \widetilde{S}(t) \leq \delta$, where $\widehat{S}(t)$ and $\widetilde{S}(t)$ represent the solution to the problem (17) obtained by COCA and the optimal solution to the problem (17).

**Theorem 1.** *Suppose the carbon emission rate $C_j(t), \forall j \in \mathcal{D}$ is i.i.d over time slots, for any control parameter $V > 0$ (the stability-cost tradeoff parameter defined in Section 4.1), implementing the COCA algorithm under a minimization error $\delta$ of the minimization Problem (17) can achieve the following performance guarantee*

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1}\sum_{j=1}^{N} \mathbb{E}\{E_j(t)P_j(t)\} \leq P^* + \frac{B+\delta}{V}, \tag{18}$$

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1}\sum_{j=1}^{N} \mathbb{E}\{Q_j(t)\} \leq \frac{B+\delta+VP^*}{\epsilon}. \tag{19}$$

*where $P^*$ is the optimal solution to the optimization problem (10), representing the theoretical lower bound of the time-averaged electricity cost, $\epsilon > 0$ is a constant which represents the distance between the time-averaged carbon emission achieved by some stationary control strategy and the carbon emission budget, and $B$ is a finite constant parameter defined in Lemma 1.*

**Remark 4.** The theorem demonstrates an $[O(1/V), O(V)]$ cost-emission tradeoff. By using the COCA algorithm with an arbitrarily lager $V$, we can make the time-averaged electricity cost arbitrarily close to the optimum $P^*$ while maintaining the emission budget, as virtual queues $Q_j(t), \forall j \in \mathcal{D}$ are stable according to Eq. (19). Such cost reduction is achieved at the cost of a larger emission, as Eq. (19) implies that the time-averaged queue backlog grows linearly with $V$. If the emission budget $(C_1, C_2, \ldots, C_N)$ is too tight, i.e., it may be insufficient to serve all the requests. In this case, COCA will strive to reduce the actual emission as much as possible in a best-effort manner while serving all of the requests. Also, COCA can be extended to strictly enforce the emission budget by denying an appropriate amount of requests [37], which is known as request admission control. Interested readers are referred to our Appendix B, available in the online supplemental material, for a complete proof of Theorem 1.

Also note that Theorem 1 does not depend on the assumption of the long-term cost-emission tradeoff which indeed has been empirically verified in Section 2. This observation is particularly interesting: as datacenter demand response [38] is a promising approach for mitigating operational instability faced by power grids, datacenters can receive discounted electricity price if they response to the smart grids. Under this scenario, the long-term cost-emission tradeoff verified in Section 2 may not hold for geo-distributed datacenters. Fortunately, since COCA does not assume the cost-emission tradeoff, it is still applicable when datacenters participate in demand response programs.

Theorem 1 further indicates that, when with inaccurate minimization of the problem (17), we can set $V$ to a larger value to obtain the same time-averaged electricity cost as with accurate minimization. Thus, instead of making excessive effort to accurately minimize the problem (17), we propose a practically efficient decomposition using a Generalized Benders Decomposition (GBD) method [6], [39], which minimizes the problem (17) with an error $\delta$ that can be arbitrary close to 0. The detailed methodology is presented in Section 4.3.

### 4.3 Solving Problem (17) with Generalized Benders Decomposition

For better scalability and performance in terms of time complexity, it is desirable that the problem (17) can be solved in a distributed manner. A classic approach to distributed algorithm design is the dual decomposition [40] method that decomposes the problem into many sub-problems. Recently, the alternating direction method of multipliers (ADMM) has been demonstrated to be powerful in developing distributed geo-graphical load balancing algorithm [29]. Unfortunately, both dual decomposition and ADMM require the objective to be (strictly) convex, the non-convexity of problem (17) rules out the direct application of these methods. The difficulty can be further observed from the product of the decision variables $m_j(t)$ and $\mu_j^v(t)$ in the

objective, which disables the decomposition of them. Instead, we propose a centralized algorithm, with trace-driven simulations in Section 5.5, we shown that it is computationally efficient, with tens of iterations to converge and a running time less than $0.2$ second.

The transformed problem (17) is still a non-convex problem, mixed with *complicating* [41] variable vector $m$. When $m$ is temporarily held fixed, it renders the original problem (17) a convex problem, which is much easier to solve. Moreover, the projection of problem (17) onto $m$ is a linear problem (LP), and can be obtained explicitly using non-linear duality theory and relaxation technique. Fortunately, these features of the problem (17) can be fully utilized by Generalized Benders Decomposition [6], [39] method.

In this section, an efficient decomposition algorithm based on Generalized Benders Decomposition method is provided. We first analyze the derivation of the master problem, i.e., the product of the Generalized Benders Decomposition, and then focus on the key technique for solving the master problem. The GBD procedure for the original problem is given at the end of this section.

#### 4.3.1 Derivation of the Master Problem

Since our focus is to solve problem (17) at each single time slot $t$, for convenience, we would omit the variable $t$ in the following problem formulation. Due to space limit, the details for deriving the master problem MGBD with GBD method is omitted here, interested readers are referred to Appendix C, available in the online supplemental material, for a complete description of the derivation. The obtained master problem is given as follows.

*Problem MGBD.* Using the transformations $\mathcal{V}$ and $v$ shown in Appendix C, available in the online supplemental material, and taking the definition of supreme as the least upper bound, we obtain the *master problem* MGBD:

$$\min_{m\in Y, m_0} \quad m_0, \tag{20}$$

$$\text{s.t.} \quad m_0 \geq L^*(m;\varphi), \quad \forall \varphi \geq 0, \tag{21}$$

$$L_*(m;\lambda) \leq 0, \quad \forall \lambda \in \Lambda, \tag{22}$$

where $L^*(m;\varphi) \equiv \inf_{R,\mu\in X} f(m,R,\mu) + \sum_{j=1}^{N} \varphi_j g_j(m,R,\mu)$, $L_*(m;\lambda) \equiv \inf_{R,\mu\in X}[\sum_{j=1}^{N} \lambda_j g_j(m,R,\mu)]$.

We will show later that the master problem MGBD satisfies *property P*: for each $\varphi \geq 0$, the infimum of $f(m,R,\mu) + \sum_{j=1}^{N} \varphi_j g_j(m,R,\mu)$ over $X$ can be taken independently of $m$, so that the function $L^*(m;\varphi)$ on $Y$ can be obtained explicitly with little or no more effort than is required to evaluate it at a single value of $m$; the same goes for $L_*(m;\lambda)$.

#### 4.3.2 Solving the Master Problem

The most natural strategy for solving the master problem MGBD is relaxation, since MGBD has a very large number of constraints. Begin by solving a relaxed version of master problem that ignores all but a few of the constraints (21) and (22); if the resulting solution does not satisfy all of the ignored constraints, then generate and add to the relaxed problem one or more violated constraints and solve it again; continue in this fashion until a relaxed problem solution which satisfies all of the ignored constraints has been obtained.

As mentioned in Section 4.3.1, the subproblem CVP($\hat{m}$) can be used to test the feasibility of a solution to a relaxed version of MGBD with respect to the ignored constraints, and to generate an index $\overline{\varphi}$ of a violated constraint in the event of infeasibility. If CVP($\hat{m}$) is infeasible, we can design a feasible-check problem CVPF($\hat{m}$) [6] corresponding to it which generates an index $\overline{\lambda}$ of a violated constraint. Due to space limit, the detailed dual-based solution for the master problem MGBD is omitted here, interested readers are referred to Appendix D, available in the online supplemental material, for a complete description of the proposed solution.

#### 4.3.3 Statement of the GBD Based Algorithm for MGBD

The Generalized Benders Decomposition procedure can now be stated formally. It can be easily shown that $\mathcal{V}$ has a representation in terms of a finite collection of constraints, so our algorithm can achieve a finite $\delta$-convergence [39], which means for any given error $\delta > 0$, the Generalized Benders Decomposition procedure terminates in a finite number of steps.

**Algorithm 2.** GBD based Algorithm for MGBD

**Step (0).** Let a point $\overline{m} \in Y$ be known, and select the convergence tolerance parameter $\delta > 0$. Solve the subproblem CVP($\overline{m}$).

**Step (0A).** If the subproblem CVP($\overline{m}$) is feasible, obtain the optimal solution $(\overline{R},\overline{\mu};\overline{\varphi})$ and the function $L^*(m;\overline{\varphi})$. Put $p=1$, $q=0$, and $\varphi^1 = \overline{\varphi}$.

**Step (0B).** If the subproblem CVP($\overline{m}$) is infeasible, solve the problem CVPF($\overline{m}$), and obtain the optimal solution $(\overline{R},\overline{\mu};\overline{\varphi})$ as well as the function $L_*(m;\overline{\lambda})$. Put $p=0$, $q=1$, and $\lambda^1 = \overline{\lambda}$.

**Step (1).** Solve the current relaxed master problem

$$\begin{aligned} \min_{m\in Y, m_0} \quad & m_0, \\ \text{s.t.} \quad & m_0 \geq L^*(m;\varphi^k), \quad k=1,\ldots,p, \\ & L_*(m;\lambda^k) \leq 0, \quad k=1,\ldots,q, \end{aligned}$$

by any applicable LP algorithm. Let $(\hat{m},\hat{m}_0)$ be an optimal solution; $\hat{m}_0$ is an lower bound on the optimal value of MGBD. Put $LB = \hat{m}_0$.

**Step (2).** Solve the revised subproblem CVP($\hat{m}$).

**Step (2.A).** If CVP($\hat{m}$) is feasible, solve it and obtain the optimal solution $(\hat{R},\hat{\mu};\hat{\varphi})$ as well as the function $L^*(m;\hat{\varphi})$. If $f(\hat{m},\hat{R},\hat{\mu}) \leq LB + \delta$, terminate and obtain the optimal solution $(\hat{m},\hat{R},\hat{\mu})$. Otherwise, increase $p$ by 1, and put $\varphi^p = \hat{\varphi}$; return to Step (1).

**Step (2.B).** If CVP($\hat{m}$) is infeasible, solve the problem CVPF ($\hat{m}$), and obtain the dual optimal multiplier $\hat{\lambda}$ as well as the function $L_*(m;\hat{\lambda})$. Increase q by 1, and put $\lambda^q = \hat{\lambda}$. Return to Step (1).

## 5 Performance Evaluation

In this section, we conduct numerical studies to evaluate the performance of our carbon-aware online control algorithm. Our trace-driven simulations are based on real-world workload traces, electricity price data, and electricity generation

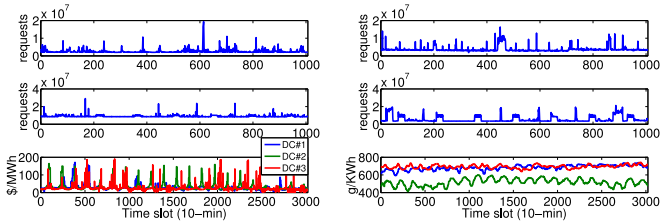



Fig. 6. The total workload trace, and the electricity price trace, carbon emission rate trace at each datacenter.

data. To fully exploit the temporal diversity of both electricity price and carbon footprint, while still reducing the overhead of switching the servers ON/OFF frequently, we use 10-min time slots as in [42].

### 5.1 Simulation Setup

We simulate a cloud with $M = 4$ front-end proxy servers located at Oregon, Iowa, Pennsylvania, and Florida. We deploy $N = 3$ datacenters in three locations in North America: California, Alberta[1] and Oklahoma. We now describe the real-world data sets and system parameters in more details.

*Workload data*. In this simulation, the workload we use is a set of traces taken from four RAID volumes of an enterprise storage cluster in Microsoft [15]. The trace includes the timestamp, hostname, disknumber, etc. Specifically, we use the trace of each RAID volume to represent the workload of each front-end proxy server, and we can calculate the arrival rate $A_i(t)$ at each time slot according to the timestamp information. The workload traces are plotted in the first 4 subfigures of Fig. 6. To evaluate the long-term effectiveness of COCA, we repeat the original one-week trace to get a 3-week workload trace.

*Electricity Price Data*. We download the locational marginal prices (LMP, in unit of $/MWh) in real-time electricity markets of the three locations from the regional transmission organization or independent system operator (ISO) website. Specifically, the LMP for California and Alberta is hourly, while the LMP for Oklahoma is 5-min data. Based on this data, we obtain the average electricity price over each time slot with a 10-minute interval. The time period of this data is from August 1, 2012 to August 21, 2012, including three weeks or 3,024 time slots. This trace is plotted in the fifth subfigure of Fig. 6, and one can observe that DC#3 (Oklahoma) enjoys a relatively cheaper price.

*Electricity generation data*. To estimate the carbon emission rate of each datacenter location, we first download the electricity generation data from the RTO or ISO website. They report the hourly electricity fuel mix for generating electricity. We then calculate the hourly carbon emission rates (in unit of g/KWh) of the three locations according to Eq. (8) given in Section 3.4. The time period of this data is also from August 1, 2012 to August 21, 2012. This trace is shown in the last subfigure of Fig. 6. It is clear that the electricity generated in California (DC#2) is "greener" than that of the other two regions.

1. Currently, most of the Independent System Operators in America do not release fine time-scale (i.e., hourly) electricity generation data on their websites. In order to fully diversify the carbon emission rates of the sampled datacenters, we place a datacenter in Alberta, Canada, where the hourly electricity generation data is assessable.

TABLE 3
Server Parameters in Three Datacenters

| Location | $s_j$ (requests/s) | $\alpha_j$ | $\beta_j$ (Watt) | $M_j$ (servers) |
|---|---|---|---|---|
| Alberta | 20 | 0.3 | 120 | 1,250 |
| California | 25 | 0.2 | 125 | 1,800 |
| Oklahoma | 20 | 0.25 | 100 | 1,500 |

*System parameters*. We first choose a high energy efficiency level, i.e., $\text{PUE}_j = 1.3$, for all three datacenters. We also choose a typical setting [7] of exponent parameter $\nu_j = 2$. The other server parameters at each datacenter are presented in Table 3. We calculate the round-trip network latency $L_{ij}$ according to the empirical approximation $L_{ij} = 0.02\,\text{ms/km} \times d_{ij}$. The geographical distance $d_{ij}$ can be obtained via mapping applications such as Google Maps. The average network latency perceived by each front-end proxy server $i$ is enforced to be within 50 ms, thus, $L_i^{\max} = 0.05$ s, $\forall i \in \mathcal{S}$.

*Platform*. We conduct the numerical study on Matlab2014R, which runs on a Intel Xeon E5-2,670 server with 8-core CPU (2.6G) and 8 GB DDR3 memory. The server runs Linux 2.6.32 kernel.

### 5.2 Performance Benchmark

To analyze the performance improvement of our COCA framework, and set an appropriate carbon emission budget for each datacenter, we use the following benchmark schemes that represent the two extreme tradeoff points between cost minimization and emission minimization: (1) Carbon-oblivious scheme (COS) [6], which only minimizes the electricity cost at each time slot, without considering the carbon emission; and (2) Electricity-oblivious scheme (EOS), which solely focuses on carbon emission minimization.

The overall electricity cost and time-averaged carbon emission of EOS and COS are shown in Fig. 7. We observe that: (1) Under EOS, DC#2 (California) dominantly outputs 68.5 percent of the total carbon emission and consumes 79.4 percent of the total electricity cost. This is because the carbon emission rate in California is much lower than other regions (revealed in Fig. 6), and it attracts more workload. (2) Similarly, under COS, DC#3 (Oklahoma) chiefly contributes 60.6 percent of the total electricity cost and emits 65.0 percent of the total emission, as Oklahoma provides the cheapest electricity price. (3) When compared with EOS, COS consumes 43.7 percent less electricity cost, at the expense of producing 16.4 percent more carbon emission.

### 5.3 Cost-Emission Tradeoff

Intuitively, total carbon emission can be reduced by migrating workload from DC#1 and DC#3 to DC#2, since DC#2

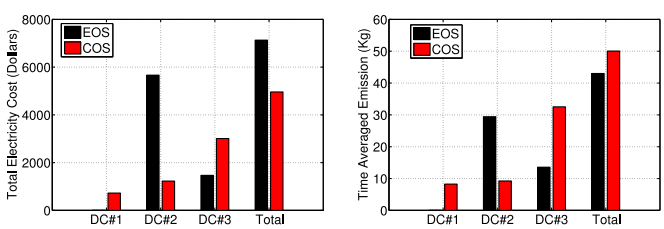



Fig. 7. Total electricity cost and time-averaged carbon emission of each datacenter under electricity-oblivious scheme (EOS) and carbon-oblivious scheme (COS).

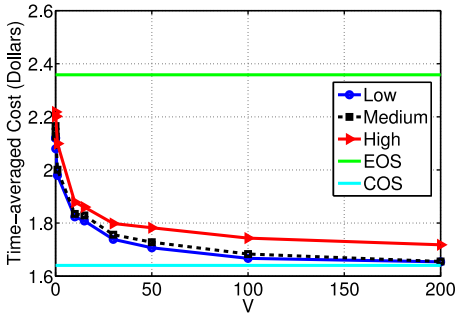



Fig. 8. Time-averaged electricity cost versus different values of the control parameter $V$ under different emission configuration.

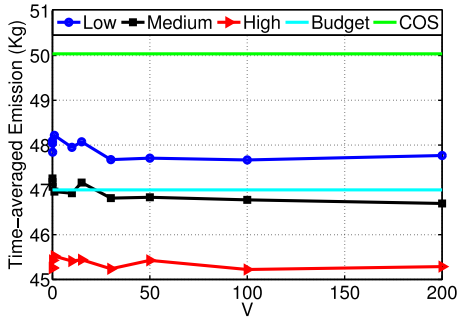



Fig. 10. Time-averaged carbon emission versus different values of the control parameter $V$ under different emission configuration.

enjoys the greenest electricity supply. In this section, we evaluate the cost-emission tradeoff achieved by COCA under different configurations of workload migration. Specifically, we first set the time-averaged carbon reduction target to be $3.04$ Kg, meaning that the total time-averaged emission budget of the three datacenters is equal to $47$ Kg. Then, we set three different migration configurations corresponding to different amounts of workload migrated to DC#2: Low $(6, 13, 28)$ Kg, Medium $(5, 16, 26)$ Kg, High $(4, 20, 23)$ Kg.

*Optimality of electricity cost*. Fig. 8 plots the time-averaged electricity cost for different values of the control parameter $V$ in our COCA algorithm under various emission configurations. We make the following observations. First, as $V$ increases, cost achieved by COCA decreases significantly and converges to the minimum level. This quantitatively confirms Theorem 1 in that COCA can approach the optimal cost with a diminishing gap $(1/V)$ (captured by Eq. (18)). The cost reduction diminishes as $V$ increases, however, as cost will eventually achieve the minimum. Second, as a comparison, we plot the time-averaged electricity cost under COS and EOS. Fig. 8 shows that cost achieved by COCA is always between that achieved by COS and EOS, and gets closer and closer to that achieved by COS as $V$ increases. This demonstrates that COCA is cost-effective when reducing carbon emission. Third, comparing electricity cost under different emission configurations, we find that the more emission migrated to DC#2, the higher the cost would be, since DC#2 is the most expensive region.

*Queue stability*. Fig. 9 plots the total time-averaged queue backlog for different values of $V$ under various emission configurations. As $V$ increases, the total backlog increases almost linearly, which is captured by Eq. (19) in Theorem 1. Along with Fig. 8, this reflects the tradeoff between queue stability and cost minimization. Further, when $V$ is relatively small, i.e., the emphasis is on emission budget enforcement, migrating emission to DC#2 alleviates the congestion of the virtual queuing system simply because DC#2 is most effective in reducing carbon emission.

*Carbon emission*. Fig. 10 plots the total time-averaged carbon emission of the three datacenters for various values of $V$ under different emission configurations. Observe that though carbon emission under the tightest budget configuration – Low emission budget configuration – is not enforced within the budget as $V$ changes, it is still far below the emission of EOS. These demonstrate that COCA would strive to reduce the emission as much as possible, which has been discussed in the remark of Theorem 1.

In order to avoid the case that an emission configuration is too tight to serve all requests, and to show the effectiveness of our COCA framework, we redefine $Q(t)$ to limit the emission of the cloud rather than that of each datacenter. That is,

$$Q(t+1) = \max\left[Q(t) - C + \sum_{j=1}^{N} E_j(t)C_j(t),\ 0\right],$$

where $C$ represents the emission budget of the cloud, and $Q(t)$ is the historical measurement of the backlog between the total time-averaged emission during the interval $[0, t-1]$ and the cloud's emission budget $C$. Then, the objective of COCA is now transformed to minimize the term $\sum_{j=1}^{N} E_j(t)[VP_j(t) + Q(t)C_j(t)]$ at each time slot.

*Electricity cost versus carbon emission*. To explore the delicate tradeoff between the electricity cost and the actual carbon emission, we vary the emission budget $C$ from $43$ to $50$ Kg, with a step size of $1$ Kg. Fig. 11 illustrates the tradeoff curves under different values of $V$. We observe the following interesting trends. First, for a given $V$, the actual emission of the cloud decreases with the decline of emission budget $C$.

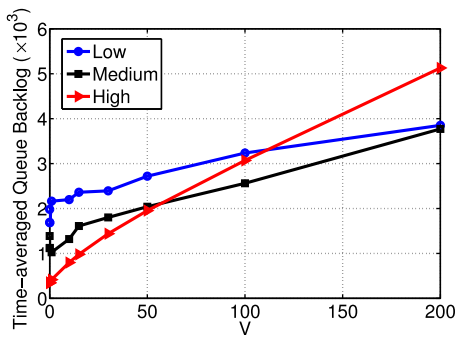



Fig. 9. Time-averaged queue backlog versus different values of the control parameter $V$ under different emission configuration.

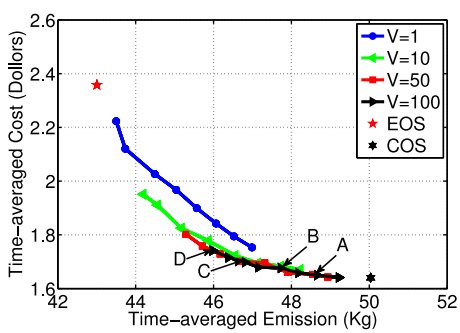



Fig. 11. Time-averaged electricity cost versus time-averaged carbon emission under different values of the control parameter $V$.

TABLE 4
Changes of Performance Brought by COCA

| Point | A | B | C | D |
|---|---|---|---|---|
| Cost Rising (%) | 0.57 | 2.20 | 3.53 | 6.17 |
| Emission Reduction (%) | 2.75 | 4.61 | 6.44 | 8.11 |

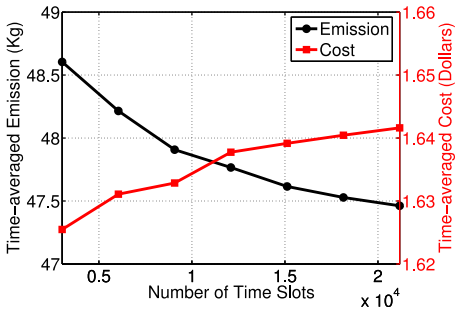



Fig. 12. Time-averaged electricity cost and carbon emission versus the length of simulated period, with $C = 47\,\mathrm{Kg}$, $V = 900$.

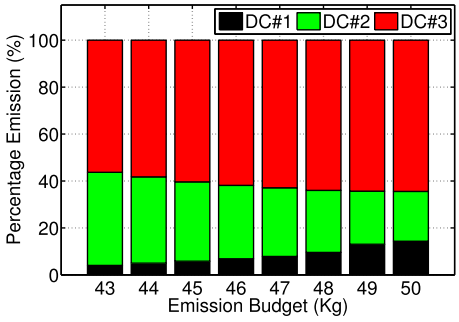



Fig. 13. Percentage of carbon emission of each datacenter under different emission budget, with $V = 100$.

On the other hand a marked increase of the electricity cost is also incurred. Second, under the same level of actual carbon emission, a larger $V$ brings a lower cost, since the control parameter $V$ represents how much we emphasize the cost compared to the emission. Third, when comparing to COS, COCA is effective in reducing carbon emission without incurring excessive cost increase, especially under a large value of $V$. The performance changes of points A–D are plotted in Fig. 11, and more details are listed in Table 4.

*Enforcement of carbon emission.* In Fig. 11, we observe that as long as the emission budget $C$ is relatively small, the actual emission is not enforced within the budget. We demonstrate in Fig. 12 the effectiveness of the emission enforcement by tuning the length of simulated period from 3,024 time slots to 21,168 (i.e., 147 days). It clearly shows that, the actual emission diminishes significantly and gradually approaches the emission budget. In a real-world cloud, the emission reduction target is usually set out for a period of several years. Hence, COCA is efficient to enforce the emission of a practical cloud system.

### 5.4 Inside of the Cost-Emission Tradeoff

To further understand how COCA works to arbitrate the emission-cost tradeoff, we continue to explore the effect of various parameters.

*Carbon migration.* We individually tune the emission budget $C$ and the control parameter $V$, and then compare the

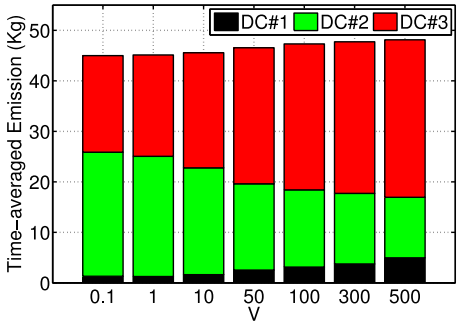



Fig. 14. Time-averaged carbon emission of each datacenter under different values of the control parameter $V$, with $C = 47\,\mathrm{Kg}$.

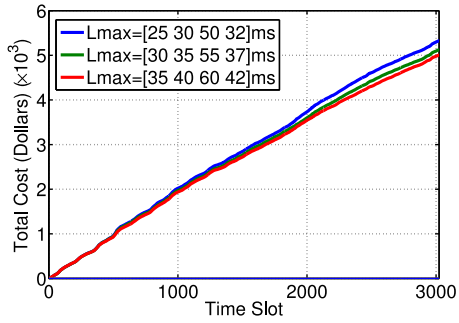



Fig. 15. Total electricity cost at each time slot under different SLA requirements, with the Medium emission configuration, and $V = 100$.

carbon emission of each datacenter. The results are illustrated in Figs. 13 and 14, respectively. Fig. 13 suggests that, when relaxing the emission budget, a larger proportion of the actual emission would be migrated from the low-carbon datacenter DC#2 to the low-price datacenters DC#1 and DC#3. We also observe from Fig. 14 that, with the increase of the control parameter $V$, more emission would be migrated from the low-carbon datacenter DC#2 to the low-price datacenters DC#1 and DC#3 to meet the stronger emphasis on cost minimization. Meanwhile, the total emission of the cloud also deteriorates as $V$ increases, demonstrating the important role of the control parameter $V$ as the design knob of the emission-cost tradeoff.

*SLA requirement.* To investigate the impact of SLA on the electricity cost, we adjust the SLA requirements while fixing other control and system parameters. We also use heterogeneous SLA requirements for different front-end proxy servers. Specifically, we choose $L^{\max} = \{[25\ 30\ 50\ 32], [30\ 35\ 55\ 37], [35\ 40\ 60\ 42]\}$ms and fix the control parameter $V = 100$, under the medium emission configuration. As expected in Fig. 15, the relaxation of the maximal tolerable network latency $L^{\max}$ gives more opportunity to cut down the total electricity cost, since a loose SLA requirement gives more feasibility to route requests to remote datacenters with cheap electricity price, and thus to reduce the electricity bill.

*Adaptive queue backlog management.* To further understand the role of queue backlog in the control of emission budget, we depict the fluctuation of the total queue backlog during the 3,024 time slots under different values of $V$ in Fig. 16. It is clear that the real-time total queue backlog fluctuates frequently. The explanation to this frequent fluctuation is that when the queue backlog is large and thus increasing the possibility of violating the emission budget, COCA will strive to reduce the queue backlog, in order to prevent the actual

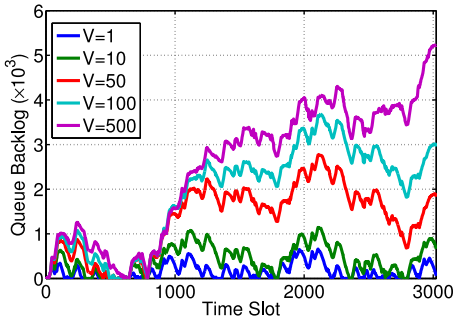



Fig. 16. Queue backlogs of datacenter under different values of the control parameter V, with the Medium emission configuration.

emission from violating the emission budget. When the queue backlog is small, COCA will primarily minimize the electricity bill, resulting in the growth of the queue backlog. A closer look at Fig. 16 also shows that the real-time total queue backlog fluctuates more frequently with a smaller value of $V$, since it puts a stronger emphasis on emission enforcement.

### 5.5 Additional Advantages of COCA

Having understood the cost-emission tradeoff, we further examine other advantages of our algorithm, including fast convergence and power proportionality.

*Convergence and running time.* The convergence speed of the inner GBD-based algorithm under different control parameter $V$ is critical when putting COCA into practical implementation. Fig. 17 plots the CDF of the number of iterations that our algorithm takes to achieve convergence for the $3{,}024$ runs under different values of $V$. Interestingly, we find that the curves under different values of $V$ are highly *overlapped*, which suggests that our GBD-based algorithm is robust to the selection of $V$. Furthermore, it clearly shows that our algorithm is able to converge within $40$ iterations for $90$ percent of the total runs. For more than half of the total runs, it can converge within $20$ iterations. The fastest run uses only $3$ iterations to converge, and all the runs converge in no more than $60$ iterations. These facts demonstrate the fast convergence of our GBD-based algorithm. We further examined the running time of COCA under different values of $V$, for $V = 1,\ 10,\ 100$ and $500$, the total running time are $527.52,\ 519.31,\ 531.77$ and $535.04$ seconds, respectively. Since there are $3{,}024$ time slots in total, solving the real-time problem at each time slot takes around $0.17$ seconds, which demonstrates the computational efficacy of COCA.

*Power proportionality.* The enormous energy demand and carbon emission of cloud computing have forced a growing

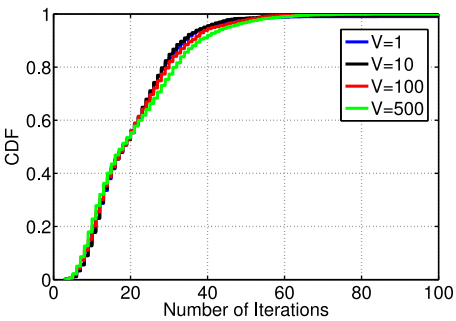



Fig. 17. Number of iterations under different values of the control parameter V, with the Medium emission configuration .

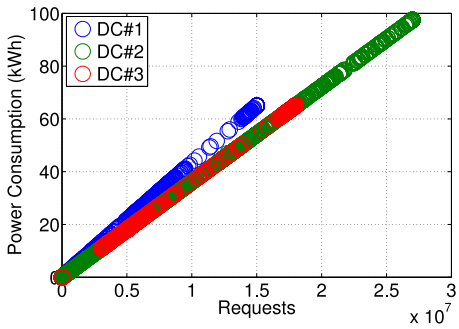



Fig. 18. Power consumption versus workload of each datacenter in each time slot.

push to improve the energy efficiency of the datacenters. A guiding focus for research into "green" data centers is the goal of designing datacenters that are "power-proportional", i.e., using power only in proportion to the workload. Excitingly, we find that, with the help of dynamic datacenter right sizing and server speed scaling, COCA could achieve an ideal power-proportionality. Fig. 18 plots the strict proportionality between power consumption and workload at each datacenter. Specifically, the proportionality factor (i.e., per-request power consumption) of each datacenter equals to the per-resquest power consumption when the respective servers in each datacenter running at full load and maximal service rate (e.g., for DC#2, the per-resquest power consumption $0.0036$ W equals to $1.3 \times 250\,\mathrm{W}/(25 \times 3{,}600) = 0.0036$ W). Besides, we also observe that the DC#2 and DC#3 are more energy efficient than the DC#1.

## 6 Related Work

The enormous energy consumption and carbon emission in datacenters have motivated extensive research [43]. Existing works have mainly focused on reducing energy consumption or electricity bill [5], [6], [7], [8], [10], which, as we have demonstrated earlier, does not necessarily translate to carbon emission reduction. Our work differs from them in that we directly examine carbon emission.

In terms of Lyapunov optimization, the concept is not new and it was applied to navigate the intuitive tradeoff between the energy cost and capacity of energy storage in [42], [44], [45]. However, the long-term cost-emission tradeoff incurred by the spatial and temporal variability of carbon footprint is non-inherent, and has not been extensively explored for geo-distributed clouds. Thus, we first take empirical studies to demonstrate that energy cost minimization conflicts with the minimization of carbon emission in realistic geo-distributed cloud services. Then, we apply Lyapunov optimization, generalized benders decomposition and convex optimization to arbitrate the long-term cost-emission tradeoff.

Our work is inspired by the pioneer work on managing carbon emission of datacenters in [2]. We complement it by considering alternative policies, namely, "Cap and Trade" and "Baseline and Credit". Moreover, in [2], the availability of power-proportional datacenters is assumed. Our framework considers multiple levels, and is able to dynamically shut off unnecessary servers and adjust CPU speed to build power-proportional datacenters. The work [2] also discusses the relation between energy cost and carbon emission from the national level, but it does not observe the

conflict between energy cost minimization and carbon emission reduction. Our work substantially complements to it, since our empirical study is based on the latest updated electricity price and emission data [18]. By further studying five representative clouds, we reveal that energy cost minimization conflicts to carbon emission reduction.

A closely related work is [46], which also targets a specified carbon emission target. Our study is different in the following aspects. First, [46] considers renewable energy capacity planning for a single datacenter to reduce emission. Our work is based on the spatial and temporal variabilities of the carbon footprint to green geo-distributed datacenters. These two complement with each other. For example, our framework can be extended to incorporate the use of renewable energy [47] to further green the cloud. Second, the datacenter is treated as a *black box* in [46], whereas our proposed framework, with capacity right-sizing and server speed scaling, can efficiently cut down the power consumption without violating the SLA of user requests. Third, an initial prediction of the entire future workload is needed in the earlier work, which however can be difficult to obtain, particularly for bursty and nonstationary workloads. Our framework does not rely on such a predictor, and instead makes online decisions to enforce the long-term emission budget.

## 7 Conclusion

In response to the enormous energy demand and carbon emission of geo-distributed datacenters, this paper explored the spatial and temporal variabilities of carbon footprint to cut down the carbon emission of the cloud. We first demonstrated the existence of the cost-emission tradeoff, through an empirical study on leading geo-distributed clouds. We then designed and analyzed a carbon-aware online control framework to balance the three-way tradeoff between electricity cost, SLA requirement and emission reduction budget. Applying Lyapunov optimization, our carbon-aware online control framework makes decisions dynamically across different levels, including geographical load balancing, capacity right-sizing, and server speed scaling. We proved that our control framework approaches a delicate $[O(1/V), O(V)]$ cost-emission tradeoff. It allows a cloud to achieve a time-averaged electricity cost arbitrarily close to the optimum, while maintaining the long-term carbon emission budget. Performance evaluation with empirical data demonstrates the effectiveness of our proposed framework.

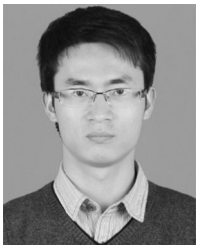
## Acknowledgments

The research was supported in part by a grant from the National Natural Science Foundation of China (NSFC) under grant No. 61520106005, by a grant from National 973 Basic Research Program under grant No. 2014CB347800. The corresponding author is Fangming Liu.

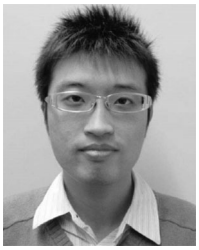
## References

[1] X. Yi, F. Liu, J. Liu, and H. Jin, "Building a network highway for big data: architecture and challenges," *IEEE Netw. Mag.*, vol. 28, no. 4, pp. 5–13, Jul./Aug. 2014.
[2] P. X. Gao, A. R. Curtis, B. Wang, and S. Keshav, "It's Not Easy Being Green," in *Proc. ACM SIGCOMM*, 2012, pp. 211–222.
[3] (2014). Google Green. [Online]. Available: http://www.google.com/green/bigpicture/#/intro/infographics-1
[4] (2011). The Guardian. [Online]. Available: http://www.guardian.co.uk/environment/2011/sep/08/google-carbon-footprint
[5] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska, "Dynamic Right-Sizing for Power-Proportional Data Centers," in *Proc. of IEEE INFOCOM*, 2011.
[6] L. Rao, X. Li, M. D. Ilic, and J. Liu, "Distributed coordination of internet data centers under multiregional electricity markets," in *Proc. IEEE*, vol. 100, no. 1, pp. 269–282, Jan. 2012.
[7] Y. Yao, L. Huang, A. Sharma, L. Golubchik, and M. J. Neely, "Data centers power reduction: A two time scale approach for delay tolerant workloads," in *Proc. IEEE INFOCOM*, 2012, pp. 1431–1439.
[8] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. H. Andrew, "Greening geographic load balancing," in *Proc. ACM ACM SIGMETRICS Joint Int. Conf. Meas. Modeling Comput. Syst.*, 2011, pp. 233–244.
[9] Z. Zhou, F. Liu, H. Jin, B. Li, and H. Jiang, "On arbitrating the power-performance tradeoff in SaaS clouds," in *Proc. IEEE INFOCOM*, 2013, pp. 872–880.
[10] A. Wierman, L. Andrew, and A. Tang, "Power-aware speed scaling in processor sharing systems," in *Proc. IEEE INFOCOM*, 2009, pp. 2007–2015.
[11] H. Xu, C. Feng, and B. Li, "Temperature aware workload management in geo-distributed datacenters," in *Proc. ACM SIGMETRICS/Int. Conf. Meas. Modeling Comput. Syst.*, 2013, pp. 373–374.
[12] (2013). Federal Energy Regulatory Commission. [Online]. Available: https://www.ferc.gov/
[13] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource Allocation and cross-layer control in wireless networks," *Found. Trends Netw.*, vol. 1, no. 1, pp. 1–149, 2006.
[14] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. San Mateo, CA, USA: Morgan Kaufmann, 2010.
[15] D. Narayanan, A. Donnelly, and A. Rowstron, "Write off-loading: Pratical power management for enterprise storage," in *Proc. USENIX Conf. File Storage Technol.*, 2008, pp. 253–267.
[16] Z. Zhou, F. Liu, Y. Xu, R. Zou, H. Xu, J. Lui, and H. Jin, "Carbon-aware load balancing for geo-distributed cloud services," in *Proc. IEEE 21st Int. Symp. Modeling, Anal. Simul. Comput. Telecommun. Syst.*, Aug. 2013, pp. 232–241.
[17] (2013). U.S. Energy Information Administration. [Online]. Available: http://www.eia.gov
[18] (2012). Electric Power Annual. (2012). [Online]. Available: http://www.eia.gov/electricity/annual/?src=Electricity-f4
[19] (2014). Google Datacenter Locations. [Online]. Available: http://www.google.com/about/datacenters/inside/locations/index.html
[20] (2014). Global Datacenters. [Online]. Available: https://www.microsoft.com/en-us/server-cloud/cloud-os/global-datacenters.aspx
[21] (2014). Global Infrastructure. [Online]. Available: https://aws.amazon.com/about-aws/global-infrastructure/
[22] (2014). The Facebook Data Center FAQ. [Online]. Available: http://www.datacenterknowledge.com/the-facebook-data-center-faq/
[23] (2014). The Apple Data Center FAQ. [Online]. Available: http://www.datacenterknowledge.com/the-apple-data-center-faq/
[24] (2015). The Green Grid. [Online]. Available: http://www.thegreengrid.org
[25] (2013). Googles Green PPAs: What, How, and Why. [Online]. Available: www.google.com/green/pdfs/renewable-energy.pdf
[26] (2014). Microsoft Environment. [Online]. Available: www.microsoft.com/environment
[27] (2014). Facebook Environment. [Online]. Available: www.facebook.com/iloveenvironment
[28] Y. Chen, R. Mahajan, B. Sridharan, and Z. Zhang, "A provider-side view of web search response time," in *Proc. ACM SIGCOMM*, 2013, pp. 243–254.
[29] H. Xu and B. Li, "Joint request mapping and response routing for geo-distributed cloud services," in *Proc. IEEE INFOCOM*, 2013, pp. 854–862.
[30] A. Qureshi, "Power-demand routing in massive geo-distributed systems," Ph.D. dissertation, Massachusetts Instit. Technol., Cambridge, MA, USA, 2010.
[31] S. M. Rumble, D. Ongaro, R. Stutsman, M. Rosenblum, and J. Ousterhout, "It's time for low latency," in *Proc. 13th USENIX Conf. Hot Topics Oper. Syst.*, 2011, p. 11.
[32] Y. Xiang, T. Lan, V. Aggarwal, and Y. Chen, "Joint latency and cost optimization for erasure-coded data center storage," in *arXiv 1404.4975*, 2014.

[33] M. Szymaniak, D. Presotto, G. Pierre, and M. van Steen, "Practical large-scale latency estimation," *Comput. Netw.*, vol. 52, no. 7, pp. 1343–1364, 2008.
[34] (2013). ISO Express. [Online]. Available: http://isoexpress.iso-ne.com/guest-hub
[35] L. Zhang, C. Wu, Z. Li, C. Guo, M. Chen, and F. Lau, "Moving big data to the cloud: an online cost-minimizing approach," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 12, pp. 2710–2721, Dec. 2013.
[36] Y. Jin and J. Branke, "Evolutionary optimization in uncertain environments-a survey," *IEEE Trans. Evol. Comput.*, vol. 9, no. 3, pp. 303–317, Jun. 2005.
[37] M. J. Neely, "Delay-based network utility maximization," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[38] Z. Zhou, F. Liu, Z. Li, and H. Jin, "When smart grid meets geo-distributed cloud: An auction approach to datacenter demand response," in *Proc. IEEE INFOCOM*, 2015, pp. 2650–2658.
[39] A. M. Geoffrion, "Generized benders decomposition," *J. Optimization Theory Appl.*, vol. 10, no. 4, pp. 237–260, 1972.
[40] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[41] (2015). Primal and Dual Decomposition Notes. [Online]. Available: http://www.stanford.edu/class/ee364b/lectures.html
[42] Y. Guo and Y. Fang, "Electricity cost saving strategy in data centers by using energy storage," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1149–1160, Jun. 2013.
[43] Z. Zhou, F. Liu, B. Li, B. Li, H. Jin, R. Zou, and Z. Liu, "Fuel cell generation in geo-distributed cloud services: A quantitative study," in *Proc. IEEE 34th Int. Conf. Distrib. Comput. Syst.*, 2014, pp. 52–61.
[44] R. Urgaonkar, B. Urgaonkar, M. J. Neely, and A. Sivasubramaniam, "Optimal power cost management using stored energy in data centers," in *Proc. ACM SIGMETRICS Joint Int. Conf. Meas. Modeling Comput. Syst.*, 2009, pp. 221–232.
[45] W. Deng, F. Liu, X. Liu, and H. Jin, "MultiGreen: Cost-minimizing multi-source datacenter power supply with online control," in *Proc. ACM e-Energy*, 2013, pp. 149–159.
[46] C. Ren, D. Wang, B. Urgaonkar, and A. Sivasubramaniam, "Carbon-aware energy capacity planning for datacenters," in *Proc. IEEE 20th Int. Symp. Modeling, Anal. Simul. Comput. Telecommun. Syst.*, 2012, pp. 391–400.
[47] W. Deng, F. Liu, H. Jin, B. Li, and D. Li, "Harnessing renewable energy in cloud datacenters: Opportunities and challenges," *IEEE Netw. Mag.*, vol. 28, no. 1, pp. 48–55, Jan./Feb. 2014.



**Zhi Zhou** received the BS and ME degrees from the School of Computer Science and Technology, Huazhong University of Science and Technology (HUST), Wuhan, China. He is currently working toward the PhD degree in the School of Computer Science and Technology, HUST. His primary research interests include green cloud computing and smart grid.



**Fangming Liu** received the BEngr degree in 2005 from the Department of Computer Science and Technology, Tsinghua University, Beijing; and the PhD degree in computer science and engineering from the Hong Kong University of Science and Technology, Hong Kong, in 2011. He is a full professor in the Huazhong University of Science and Technology, Wuhan, China. His research interests include cloud computing and datacenter, mobile cloud, green computing, SDN, and virtualization. He is selected into National Youth Top Talent Support Program of National High-level Personnel of Special Support Program (The "Thousands-of-Talents Scheme") issued by Central Organization Department of CPC. He is a youth scientist of National 973 Basic Research Program Project of SDN-based Cloud Datacenter Networks. He was a StarTrack visiting faculty in Microsoft Research Asia in 2012 to 2013. He has been the editor-in-chief of *EAI Endorsed Transactions on Collaborative Computing*, a guest editor for *IEEE Network Magazine*, an associate editor for *Frontiers of Computer Science*, and served as a TPC for ACM Multimedia 2014, e-Energy 2016, IEEE INFOCOM 2013-2016, ICNP 2014, IWQoS 2016, and ICDCS 2015-2016. He is a member of the IEEE.

**Ruolan Zou** received the BS degree from the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China, in 2014. She is currently working toward the MS degree in the School of Computing Science at Simon Fraser University, British Columbia, Canada. Her primary research interests include datacenter, networking, and cloud computing.

**Jiangchuan Liu** (S'01-M'03-SM'08) received the BEng degree (cum laude) from Tsinghua University, Beijing, China, in 1999, and the PhD degree from The Hong Kong University of Science and Technology in 2003, both in computer science. He is a full professor in the School of Computing Science, Simon Fraser University, British Columbia, Canada, and an EMC-Endowed visiting chair professor of Tsinghua University, Beijing, China from 2013 to 2016. From 2003 to 2004, he was an assistant professor at The Chinese University of Hong Kong. He is a corecipient of ACM TOMCCAP Nicolas D. Georganas Best Paper Award 2013, ACM Multimedia Best Paper Award 2012, IEEE Globecom 2011 Best Paper Award, and IEEE Communications Society Best Paper Award on Multimedia Communications 2009. His research interests include multimedia systems and networks, cloud computing, social networking, online gaming, big data computing, wireless sensor networks, and peer-to-peer and overlay networks. He has served on the editorial boards of *IEEE Transactions on Multimedia*, *IEEE Communications Surveys and Tutorials*, *IEEE Access*, *IEEE Internet of Things Journal*, *Elsevier Computer Communications*, and *Wiley Wireless Communications and Mobile Computing*. He is a senior member of the IEEE.

**Hong Xu** received the BEng degree from the Department of Information Engineering, The Chinese University of Hong Kong, in 2007, and the MASc and PhD degrees from the Department of Electrical and Computer Engineering, University of Toronto. He joined the Department of Computer Science, City University of Hong Kong in August 2013, where he is currently an assistant professor. His research interests include data center networking, cloud computing, network economics, and wireless networking. He received an Early Career Scheme Grant from the Research Grants Council of the Hong Kong SAR, 2014. He also received the best paper award from ACM CoNEXT Student Workshop 2014. He is a member of ACM and the IEEE.

**Hai Jin** received his PhD degree in computer engineering from HUST in 1994. He is a Cheung Kung Scholars chair professor of computer science and engineering at the Huazhong University of Science and Technology (HUST), China. He was awarded the Excellent Youth Award from the National Science Foundation of China in 2001. He is the chief scientist of ChinaGrid, the largest grid computing project in China, and chief scientist of the National 973 Basic Research Program Project of Virtualization Technology of Computing Systems. His research interests include computer architecture, virtualization technology, cluster computing and grid computing, peer-to-peer computing, network storage, and network security. He is a senior member of the IEEE and a member of the ACM.