Carbon-aware Load Balancing for Geo-distributed Cloud Services

Zhi Zhou[1] Fangming Liu*[1] Yong Xu[1] Ruolan Zou[1] Hong Xu[2] John C.S. Lui[3] Hai Jin[1]
[1]Key Laboratory of Services Computing Technology and System, Ministry of Education,
School of Computer Science and Technology, Huazhong University of Science and Technology, China.
[2]Department of Electrical and Computer Engineering, University of Toronto, Canada.
[3]The Chinese University of Hong Kong.

***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 the 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. We 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 analyses and real-world trace-driven empirical evaluation demonstrate its effectiveness in both minimizing electricity cost and reducing carbon emission.**

I. Introduction

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

Intuitively, as most previous works [4]–[9] have shown, carbon emission reduction may be achieved by cutting down the energy consumption. In general these approaches fall into three categories. *Geographical load balancing* [4] utilizes the server heterogeneity of geo-distributed datacenters to distribute requests so that the energy cost can be reduced. *Capacity right-sizing* [7] dynamically turns off/on redundant servers when demand decreases/increases. Finally, *server speed scaling* [9] is a technique that adjusts the CPU frequency and thus service rate of a server in order to save its energy consumption.

However, for geographically distributed datacenters, energy savings may not necessarily imply carbon emission reduction. This is because of the spatial and temporal variabilities of the electricity carbon footprint, as illustrated in Fig. 1. For one, different regions generate electricity with their respective fuel mixes, and have different carbon footprints. For another, the time-varying fuel mix also leads to temporal differences in carbon footprint even for the same location. Such diversity ought to be fully exploited if we want to further green the cloud.

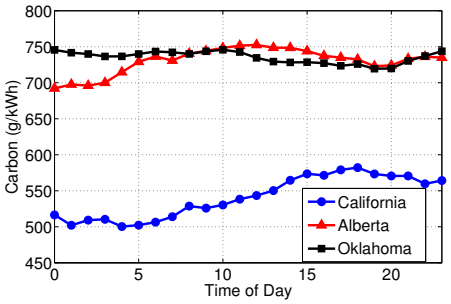

Fig. 1: Grams of carbon emission per kWh electricity at three different locations in north America on Sep 30th 2012. Data is provided by each Regional Transmission Organization (RTO) [12].

One recent work [1] proposed to reduce datacenter carbon footprint by utilizing the above variabilities. Unfortunately, it found that the current carbon tax is too low to incentivize providers to reduce carbon output, though 10% carbon emission reduction can be achieved without any cost. In reality, carbon tax is only one of the three most practical policies to inhibit carbon emission. For the other two policies —"Cap and Trade" and "Baseline and Credit", each carbon source is usually operated within an allowance or budget [10]. In practice, most of the leading providers have also announced their emission reduction budgets. For instance, Google targets to cut 30% of the carbon emission by 2020 [11]. There is a fundamental challenge when considering such an emission budget: how can we make real-time decisions to enforce the long-term emission budget without future knowledge of the bursty workload which is hard to acquire in realistic cloud? To the best of our knowledge, no previous work has considered the optimal operation of the cloud under an emission reduction budget. In this paper, we aim to bridge this gap.

Specifically, we jointly consider the electricity cost, service level agreement (SLA) requirement and emission reduction budget of geographically distributed datacenters. To navigate such a three-way tradeoff, we rigorously design and analyze a carbon-aware online control framework using *Lyanunov Optimization* [13], [14], which can effectively incorporate the long-term carbon emission constraints into real-time optimization. Our framework dynamically makes three decisions: geographical load balancing, capacity right-sizing [4], and server speed scaling [9], corresponding to the below three

*The Corresponding Author is Fangming Liu (fmliu@hust.edu.cn). The research was supported by a grant from The National Natural Science Foundation of China (NSFC) under grant No.61133006.

different levels. (1) At the service level, we determine how to distribute user requests to appropriate datacenters according to the current electricity prices and carbon emission rates; (2) at the datacenter level, we determine how many servers to activate at each datacenter; and (3) at the server level, we determine how to set the service rate of each activated server.

Given the control parameter V which represents how much we emphasize the cost minimization compared to emission enforcement, our proposed framework is rigorously proved to facilitate a delicate $[O(1/V), O(V)]$ cost-emission tradeoff, due to the fact that the greener energy is usually more expensive. With such a tradeoff, the geo-distributed cloud can achieve a time-averaged electricity cost arbitrarily close to the optimal, while still maintaining the long-term carbon emission budget. Moreover, with an empirical evaluation using the arrival rates of empirical workload traces [15] from Microsoft's enterprise storage systems, as well as real-world electricity generation and price data, we show that our proposed solution is practical to achieve a specified long-term emission reduction goal, without incurring excessive cost rising.

The rest of this paper is organized as follows. In Sec. V, we survey the related works. In Sec. II, we present the three-way tradeoff model. We further construct a carbon-aware online control framework to optimize such a three-way tradeoff and demonstrate its effectiveness in both minimizing electricity cost and enforcing carbon emission budget in Sec. III. We also evaluate the performance of our proposed framework in Sec. IV and conclude in Sec. VI.

II. The Three-way Tradeoff Model

We consider a provider running cloud services on the top of N geographically distributed datacenters, denoted by $\mathcal{D} = \{1, 2, ..., N\}$, each datacenter $j \in \mathcal{D}$ consists of M_j homogeneous servers. Although we assume that all the servers at one datacenter are homogeneous, note that our model is quite general and can be easily extended to capture the heterogeneous case with a few additional notations. The cloud deploys M front-end proxy servers, denoted by $\mathcal{S} = \{1, 2, ..., M\}$ at various regions to direct user requests to the appropriate datacenters. Inspired by the latest modeling work on datacenters [4], we consider a discrete time-slotted system where the time slot length can range from hundreds of milliseconds to minutes [8]. In every time slot $t = (0, 1, 2, ..., \tau, ...)$, user requests arrive and aggregate at each front-end proxy server. Let $A_i(t), \forall i \in \mathcal{S}$ denote the request arrival rate at front-end proxy server i during time slot t.

A. Control Decisions

Under the workload model above, we focus on *three* control decisions:

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

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

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

2) ***Datacenter right sizing****:* At the datacenter level, energy cost and carbon emission can be reduced by dynamically adjusting the number of active servers, which is known as "right sizing" [4]. Let $m_j(t)$ denote the number of servers to be activated in datacenter j at time slot t, and note that cloud datacenter typically contains thousands of active servers. Hence, the integer constraints on $m_j(t), \forall j \in \mathcal{D}, \forall t$ can be relaxed. Then, $m_j(t)$ should satisfy the following constraint:

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

3) ***Server speed scaling****:* At the server level, energy cost and carbon emission can be reduced by flexibly controlling the CPU speed of each active server, which is known as "speed scaling" [9]. Let $\mu_j(t)$ denote 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. Thus, we have:

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

B. Power Consumption Model

It has been widely demonstrated that [6], the amount of power consumed by a server running at speed μ can be characterized by $\alpha\mu^{\nu} + \beta$, where α is a positive factor, β represents the power consumption in idle state, and the exponent parameter ν is empirically determined as $\nu \geq 1$, with a typical value of $\nu = 2$ in practice [8]. Based on the above power model, given the number of active servers $m_j(t)$, parameters α_j, β_j, ν_j and the power usage efficiency metric PUE_j in datacenter j, the power consumption of datacenter j in time slot t can be quantified by $E_j(t)$ as follows:

$$E_j(t) = \text{PUE}_j \cdot m_j(t) \cdot [\alpha_j \mu_j^{\nu_j}(t) + \beta_j], \tag{5}$$

The power usage efficiency metric PUE [16] represents the ratio of the total amount of power used by the entire datacenter facility to the power delivered to the computing equipment. Inefficient datacenter facilities can have a PUE $\in [2.0, 3.0]$, while leading industry datacenter facilities are announced to approach a PUE of around 1.2 [17].

C. Real-time SLA Model

Arguably, for most cloud applications such as web search, response time is the most critical performance metric. When there is a large response delay, the provider may suffer a substantial drop in users. Therefore, the response time of a delay-sensitive service is typically enforced at an acceptable limit [18]. In this paper we focus on the average queuing delay in datacenters as it largely outweighs the network delay form a front-end proxy server to a processing datacenter [5].

To this end, we take the $M/M/n$ queuing model to estimate the response time in each datacenter. Specifically, given the request arrival rate $\sum_{i=1}^{M} R_{ij}(t)$ and service rate

$m_j(t)\mu_j(t)$ in datacenter j at time slot t, the corresponding average response time [5] can be formulated as

$$W_j(t) = \frac{1}{m_j(t)\mu_j(t) - \sum_{i=1}^{M} R_{ij}(t)}.$$

To provide good experience to users, we enforce the following constraint:

$$W_j(t) \leq W_j,$$

where W_j is the maximal tolerable response delay in datacenter j. Therefore, we have the following SLA constraint:

$$m_j(t)\mu_j(t) - \sum_{i=1}^{M} R_{ij}(t) \geq \frac{1}{W_j}. \quad (6)$$

D. Long-term Carbon Reduction Model

To characterize the spatial and temporal variability in electricities carbon emission rate, we use electricity generation data from each Regional Transmission Organization (RTO) website, where we can get 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 [19]). Thus, by summing the weighted contribution from each fuel type, we can estimate the carbon emission rate in location j at time slot t as follows:

$$C_j(t) = \frac{\sum e_{kj}(t) \times c_k}{\sum e_{kj}(t)}, \quad (7)$$

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 I.

TABLE I: Carbon dioxide emission per kilowatt-hour for the most common fuel types [1].

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

Based on the above $C_j(t)$ and aforementioned power consumption $E_j(t)$ in Eq. (5), the corresponding carbon emission is $E_j(t) \cdot C_j(t)$. In practice, as most real-world datacenters are operated within a certain carbon emission budget in a given time interval (usually one year or longer), we impose a long-term time-averaged carbon emission budget C_j for each datacenter j to reduce the carbon emission:

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\{E_j(t) \cdot C_j(t)\} \leq C_j. \quad (8)$$

E. Characterizing the Three-way Tradeoff

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

$$\min \limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{j=1}^{N} \mathbb{E}\{E_j(t)P_j(t)\}, \quad (9)$$

subject to the constraints (1), (2), (3), (4), (6), and (8).

For realistic geo-distributed datacenters, there is a potential conflict when solving this optimization problem: Since the datacenters' workload as well as the carbon emission rate is time-varying and unpredictable, 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?

III. Carbon-aware Online Control Framework

To address the challenge of the optimization problem (9), we use Lyapunov optimization techniques [14] to design a carbon-aware online control algorithm, called **COCA**, that is able to explicitly transform the long-term objective and constraints to an optimization in each time slot. In particular, **COCA** can be proved to achieve a time-averaged electricity cost arbitrarily close to optimum, while still maintaining the long-term carbon emission budget.

A. Problem Transformation Using Lyapunov Optimization

To accommodate the long-term carbon emission constraint (8), we first transform it into a queue stability problem [14]. Specifically, 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:

$$Q_j(t+1) = \max[Q_j(t) - C_j + E_j(t)C_j(t),\ 0], \quad (10)$$

where $C_j, E_j(t)$ and $C_j(t)$ are defined in Sec. II-D. $\forall j \in \mathcal{D}$, $E_j(t)C_j(t)$ can be viewed as the "arrivals" of virtual queue $Q_j(t)$, while 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, and 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 (8) 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. (10) it is clear that:

$$Q_j(t+1) \geq [Q_j(t) - C_j + E_j(t)C_j(t)].$$

By summing this inequality over time slots $t \in \{0, 1, ..., T-1\}$ 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:

$$\lim_{T\to\infty} \frac{\mathbb{E}\{Q_j(t)\}}{T} + C_j \geq \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\{E_j(t)C_j(t)\}. \quad (11)$$

If the virtual queues $Q_j(T)$ are stable, then $\lim_{T\to\infty} \mathbb{E}\{Q_j(T)\}/T = 0$ (Note that we will prove the strong stability of virtual queues $Q_j(T)$ in Theorem 1 later). Subtracting this into (11) yields the inequality (8).

1) Characterizing the Emission-Cost Tradeoff: Let $\mathbf{Q}(t) = (Q_j(t))$ denote the vector of all the virtual queues. Then, we define the Lyapunov function as follows:

$$L(\mathbf{Q}(t)) = \frac{1}{2}\sum_{j=1}^{N} Q_j^2(t). \tag{12}$$

This represents a scalar metric of *congestion* [14] of all the 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 an supremum bound on the following *drift-plus-cost* expression in each time slot:

$$\Delta(\mathbf{Q}(t)) + V\mathbb{E}\Big\{\sum_{j=1}^{N} E_j(t)P_j(t)\Big\}. \tag{13}$$

Remark: The control parameter $V(\geq 0)$ represents *a design knob [8], [20] of the emission-cost tradeoff*, *i.e.,* how much we shall emphasize the cost minimization (**Problem** (9)) compared to emission budget (Constraint (8)). It empowers datacenter operators to make flexible design choices among various tradeoff points between carbon emission and electricity cost. For example, one may prefer to incur as smaller expected cost as possible, while having to keep $\Delta(\mathbf{Q}(t))$ small to avoid exceeding the carbon emission budget.

2) Bounding Drift-Plus-Cost: To derive the supremum bound of the *drift-plus-cost* expression given in Eq. (13), 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:

$$\Delta(\mathbf{Q}(t)) \leq B - \sum_{j=1}^{N} Q_j(t)\mathbb{E}\{C_j - E_j(t)C_j(t)|\mathbf{Q}(t)\}, \tag{14}$$

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

Proof: Please refer to the Appendix A. ■

Based on Lemma 1, adding the expression $V\mathbb{E}\{\sum_{j=1}^{N} E_j(t)P_j(t)|\mathbf{Q}(t)\}$ to both sides of Eq. (14) yields an supremum bound of *drift-plus-cost* expression of the datacenter system:

$$\Delta(\mathbf{Q}(t)) + V\mathbb{E}\Big\{\sum_{j=1}^{N} E_j(t)P_j(t)|\mathbf{Q}(t)\Big\} \leq B - \sum_{j=1}^{N} Q_j(t)C_j + \sum_{j=1}^{N} \mathbb{E}\{E_j(t)[VP_j(t) + Q_j(t)C_j(t)]|\mathbf{Q}(t)\}. \tag{15}$$

B. Carbon-aware Online Control Algorithm (COCA)

Instead of directly minimizing the drift-plus-cost expression in Eq. (13) that involves unknown information $Q_j(t+1)$, our carbon-aware online control algorithm **COCA** seeks to minimize the supremum bound given above (*i.e.,* equivalent to maximizing the right-hand-side term of inequality (15)), without undermining the optimality and performance of the algorithm according to [14].

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 (15).
3) Update the queues $\mathbf{Q}(t+1)$ according to Eq. (10) and the newly determined control decisions.

Till now, we have transformed the long-term optimization (9) to the following optimization at each time slot t:

$$\min \sum_{j=1}^{N} E_j(t)[VP_j(t) + Q_j(t)C_j(t)], \tag{16}$$
$$\text{s.t. } (1), (2), (3), (4), (6) \text{ are satisfied.}$$

Remark: The transformed problem (16) embodies an economic interpretation. At each time slot t, we strive 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.

The transformed optimization problem (16) is a nonlinear problem that has been studied extensively, and there exist some practical methods developed to solve these nonlinear problems with special structures. In this work, we can solve the problem (16) by using KKT conditions [21], [22] and generalized Benders decomposition (GBD) [5], [23]. The detailed solution can be found in Appendix B.

C. Optimal Analysis

We further analyze the optimality of the above proposed online control algorithm, in terms of a tradeoff between the cost minimization and emission enforcement.

***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 Sec. III-A), implementing the above **COCA** algorithm for all time slots can achieve the following performance guarantee:

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1}\sum_{j=1}^{N}\mathbb{E}\{E_j(t)P_j(t)\} \leq P^* + \frac{B}{V}, \tag{17}$$

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1}\sum_{j=1}^{N}\mathbb{E}\{Q_j(t)\} \leq \frac{B+VP^*}{\epsilon}. \tag{18}$$

where P^* is the optimal solution to the optimization problem (9), 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: 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. (18). Such cost reduction is achieved at the cost of a larger emission, as Eq. (18) implies that the time-averaged queue backlog grows linearly with V. However, if the emission budget $(C_1, C_2, ..., C_N)$ is too tight, *i.e.*, it is insufficient to serve all the requests, **COCA** will strive to reduce the actual emission as much as possible in a best-effort manner while serving all of the requests. Further, note that **COCA** can also be flexibly extended to strictly enforce the emission budget by denying an appropriate amount of requests [24], which is known as request admission control. Interested readers are referred to our Appendix C for a complete proof of Theorem 1.

IV. Performance Evaluation

In this section, we conduct trace-driven simulations to realistically evaluate the performance of our carbon-aware online control algorithm **COCA**. Our simulations are based on real-world workload traces, electricity price data sets, and electricity generation data sets. 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 set each of the discrete time slots to be 10-min.

A. Simulation Setup

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

Workload Data: In this simulation, the real-world workload data we use is a set of I/O traces taken from four RAID volumes of a enterprise storage cluster in Microsoft [15]. The trace includes the timestamp, hostname, disknumber, etc. Specifically, we use the trace from 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. For the purpose of illustration, we merge the traces from the four front-end proxy servers into one, which does not affect the performance of **COCA** according to our analysis in Appendix B. The merged trace is plotted in the top of Fig. 2, with a peak-to-average ratio of 2.49. To evaluate the long-term effectiveness of **COCA**, we repeat the original one-week trace to get a 3-week workload trace.

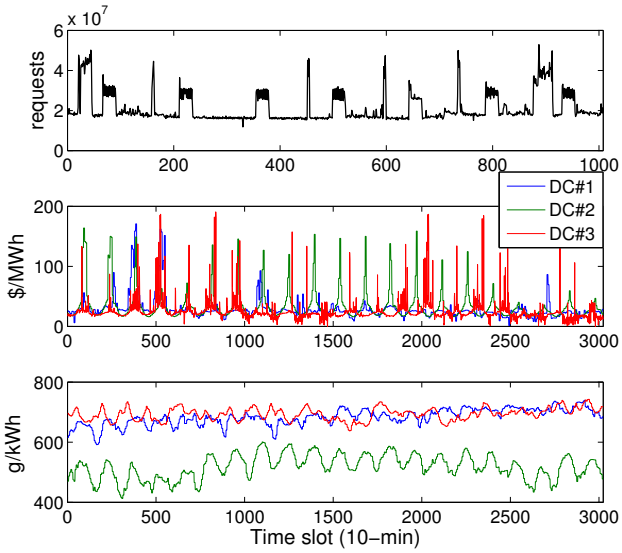

Fig. 2: The total workload trace, and the electricity price trace, carbon emission rate trace at each datacenter.

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 website of each regional transmission organization (RTO) or independent system operator (ISO). 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 Aug. 1, 2012 to Aug. 21, 2012, including three weeks or 3024 time slots. This trace is plotted in the middle of Fig. 2, and one can observe that DC#3 (Oklahoma) enjoys a relatively cheaper price when compared to the other two datacenters.

Electricity Generation Data: To estimate the carbon emission rate of each state hosting the datacenter, we first download the electricity generation data of each location from the website of the corresponding RTO or ISO. They report the hourly electricity fuel mix of each region for the major types of fuel. Then, we calculate the hourly carbon emission rates (in unit of g/KWh) of the three locations according to Eq. (7) given in Sec. II-D. The time period of this data is also from Aug. 1, 2012 to Aug. 21, 2012, the same as that of the electricity price data. This trace plotted in the bottom of Fig. 2 strongly suggests that the electricity generated in California (DC#2) has a "greener" carbon emission rate than those of the other two regions.

System Parameters: We first choose a higher energy efficiency level, *i.e.,* $\text{PUE}_j = 1.3$ for all of the three datacenters in our simulation. We also choose a typical setting [8] of exponent parameter $\nu_j = 2$ for the three datacenters. The other server parameters at each datacenter are presented in Table II. The

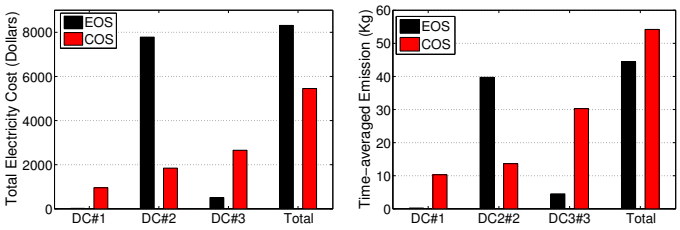

Fig. 3: Total electricity cost and time-averaged carbon emission of each datacenter under electricity-oblivious scheme (EOS) and carbon-oblivious scheme (COS).

average queuing delay at each datacenter is enforced to be within 1 ms, thus, $W_j = 0.001$s.

TABLE II: Server parameters in three datacenters.

Location	s_j (requests/s)	α_j	β_j (Watt)	M_j (servers)
Alberta	20	0.3	120	1250
California	25	0.2	125	1800
Oklahoma	20	0.25	100	1500

B. Performance Benchmark

In order 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: (i) Carbon-oblivious scheme (COS). This scheme is proposed by [5], which only minimizes the electricity cost at each time slot, without considering the carbon emission. (ii) Electricity-oblivious scheme (EOS). As the other extreme, EOS solely focuses on carbon emission minimization at each time slot, and discards the electricity cost.

The total electricity cost and time-averaged carbon emission of EOS and COS are plotted in Fig. 3. We observe that: (1) Under EOS, DC#2 (California) dominantly outputs 89.3% $(39.73/44.49)$ of the total carbon emission and consumes 93.6% $(7782.7/8312.1)$ of the total electricity cost. This is because the carbon emission rate in California is much lower than those in the other regions (revealed in Fig. 2), attracting more workload being routed to DC#2. (2) Similarly, under COS, DC#3 (Oklahoma) chiefly contributes 48.9% $(2653.5/5424.5)$ of the total electricity cost and emits 55.9% $(30.28/54.21)$ of the total emission, as Oklahoma provides the cheapest electricity price among the three regions. (3) When compared with EOS, COS consumes 52.5% $(8312.1/5452.5-1)$ less electricity cost, at the expense of producing 21.9% $(54.21/44.49-1)$ more carbon emission.

*C. Performance Evaluation of **COCA***

Intuitively, the total carbon emission of the three datacenters can be reduced by migrating some amount of workload from DC#1 and DC#3 to DC#2, since DC#2 enjoys the greenest electricity supply. In this section, we first evaluate the performance of our **COCA** framework under different configurations of workload migration. Specifically, we initially set the time-averaged carbon reduction target of the cloud to be 4.21Kg, meaning that the total time-averaged emission budget of the three datacenters is equal to 50Kg. Then, we set three different migration configurations corresponding to different amounts of workload migrated to DC#2: Low $(10, 16, 24)$Kg, Medium $(9, 20, 21)$Kg, High $(8, 25, 17)$Kg.

Optimality. Fig. 4 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 observation. *First*, as the value of V increases, the time-averaged cost achieved by **COCA** reduces significantly and converges to the minimum level for larger values of V. This quantitatively corroborates Theorem 1 in that **COCA** can approach the optimal cost with a diminishing gap $(1/V)$ (captured by Eq. (17)). However, such a cost reduction diminishes with the increase of V, as the cost will eventually achieve the minimum. *Second*, as a comparison, we plot the time-averaged electricity cost under COS and EOS. Fig. 4 shows that the time-averaged electricity 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 implies a cost-effective tradeoff between electricity cost and carbon emission as achieved by **COCA**. *Third*, when comparing the time-averaged 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 pays for the most expensive electricity.

Queue stability. Fig. 5 plots the total time-averaged queue backlog for different values of V in our **COCA** algorithm under various emission configurations. As V increases, the total time-averaged queue backlog with all configurations increases almost linearly, which is captured by Eq. (18) in Theorem 1. Along with Fig. 4, this reflects the tradeoff between queue stability and cost minimization, and the control parameter V is the design knob to tune such a tradeoff. Furthermore, it clearly shows that the more emission migrated to DC#2, the less congested the queue backlog would be, meaning a stronger enforcement of the emission budget according to the inequality (11) in Sec. III-A. This is due to the fact that DC#2 enjoys the greenest electricity supply among the three datacenters.

Carbon emission. Fig. 6 plots the total time-averaged carbon emission of the three datacenters for various values of V under different emission configurations. As we observe, the tighter emission configuration that migrates less emission budget to DC#2 actually outputs more carbon emission. Furthermore, though the emissions of the two relatively tighter configurations, low and medium configuration are not enforced within the budget as V changes, they are still far below the emission of EOS. These demonstrate that **COCA** would strive to reduce the emission as much as possible in a best-effort manner, 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 the queue $Q(t)$ so to limit the emission of the cloud rather than limit the emission of each datacenter:

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

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

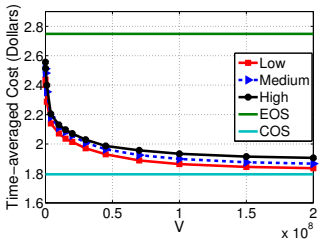

Fig. 4: Time-averaged electricity cost vs. different values of the control parameter V under different emission configuration.

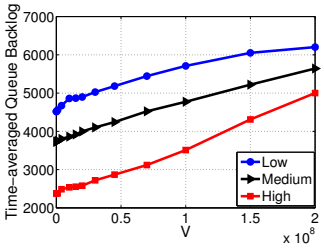

Fig. 5: Time-averaged queue backlog vs. different values of the control parameter V under different emission configuration.

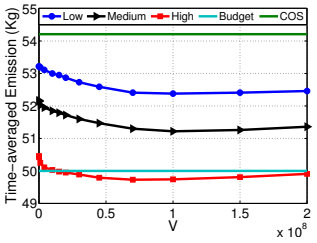

Fig. 6: Time-averaged carbon emission vs. different values of the control parameter V under different emission configuration.

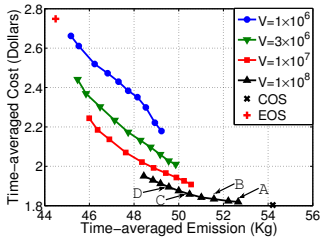

Fig. 7: Time-averaged electricity cost vs. time-averaged carbon emission under different values of the control parameter V.

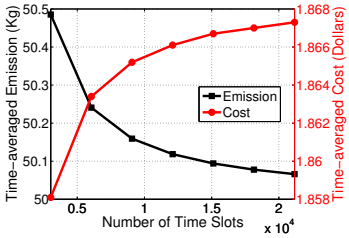

Fig. 8: Time-averaged electricity cost and carbon emission vs. the length of simulated period, with $C = 50Kg$, $V = 1 \times 10^8$.

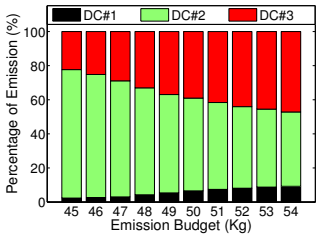

Fig. 9: Percentage of carbon emission of each datacenter under different emission budget, with $V = 1 \times 10^7$.

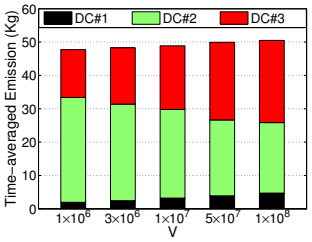

Fig. 10: Time-averaged carbon emission of each datacenter under different values of the control parameter V, with $C = 50Kg$.

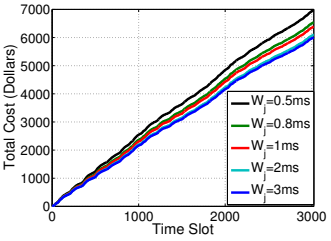

Fig. 11: Total electricity cost at each time slot under different SLA requirements, with the Medium emission configuration, and $V = 1 \times 10^7$.

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.

The emission-cost tradeoff. To explore the delicate tradeoff between electricity cost and actual carbon emission, we vary the emission budget C from 45Kg to 54Kg, with a step size of 1Kg. Fig. 7 illustrates the tradeoff curves under different values of V, we observe the following interesting trends. *First*, given the control parameter V, with the decline of emission budget C, the actual emission of the cloud also drops slightly. Such a reduction of actual emission, on the other hand, incurs a markedly increase of the electricity cost. The rationale of such a tradeoff is that under a lowered emission budget, **COCA** would migrate an appropriate amount of workload from the low-price datacenters to the low-carbon datacenters, leading to both emission reduction and cost rising. *Second*, under the same level of actual carbon emission, a larger value of V would bring a lower cost, since the control parameter V represents how much we emphasize the cost compared to the emission, as revealed in Sec. III-A. *Third*, when comparing to COS, **COCA** is effective in reducing carbon emission without incurring excessive cost increase, especially under a large value of V. Specifically, we calculate the changes of performance of the points A–D plotted in Fig. 7, and list the result in Table III.

TABLE III: Changes of Performance Brought by COCA.

Point	A	B	C	D
Cost Rising (%)	0.89	1.70	3.05	5.09
Emission Reduction (%)	2.86	4.85	6.87	8.57

Enforcement of carbon emission. In Fig. 7, we observe that as long as the emission budget C is relatively small, the actual emission is not enforced within the budget. In this section, we demonstrate the effectiveness of the emission enforcement by tuning the length of simulated period from 3024 time slots to 21168 time slots (*i.e.*, 147 days) as shown in Fig. 8. It clearly shows that, as the number of time slots grows, the actual emission diminishes significantly and gradually gets closer and closer to the emission budget, even though such a decline is at the cost of a slight increase of electricity cost. Note that in real-world cloud systems, the emission reduction target is usually carried out on a cycle of several years. Thus, the **COCA** framework is efficient to enforce the emission of a practical cloud system.

Carbon migration. To further understand how **COCA** works to arbitrate the emission-cost tradeoff, we separately tune the emission budget C and the control parameter V, then compare the carbon emission of each datacenter. The results are illustrated in Fig. 9 and Fig. 10, respectively. Specifically, Fig. 9 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 learn from Fig. 10 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. In practice, the cloud usually deploys redundant capacity to guarantee the average response delay of the workload. In this simulation, we investigate the impact of SLA requirement on the electricity cost. We adjust the SLA requirements while fixing other control and system parameters. We choose $W_j = \{0.5, 0.8, 1, 2, 3\}$ms and fix the control

parameter $V = 1 \times 10^7$, under the medium emission configuration. As illustrated in Fig. 11, the decline of the maximal tolerable response delay W_j gives more opportunity to cut down the total electricity cost, since fewer capacity needs to be deployed.

V. Related Work

The huge energy consumption and carbon emission in datacenters have motivated extensive research. However, almost all of these works focused on reducing the power consumption or electricity bill [4]–[9], [25], [26]. Our work is different from those works in that we directly study the carbon emission aspect. While recognizing the significance of the existing works [1], [27] on managing carbon emission of datacenters, our study is different from and complementary to these works.

Compared to [1] that managed the aforementioned three potentially conflicting objectives, our study differs substantially in at least two important aspects. First, we consider the alternative mechanisms —"Cap and Trade" and "Baseline and Credit"—to reduce carbon emission, which is complementary to work [1]. Second, the work [1] is based on the assumption of power-proportional datacenters, while our framework is able to dynamically shut off unnecessary servers and adjust CPU speed to build power-proportional datacenters.

Compared to [27] that also targets a specified emission reduction budget, our study is different in the following ways. First, [27] focused on renewable energy capacity planning for a single datacenter to reduce emission, while our proposed framework is based on the spatial and temporal variabilities of the carbon footprint to green the geo-distributed datacenters. Intuitively, our framework can be extended to incorporate the use of renewable energy to further green the cloud. Second, with capacity right-sizing and server speed scaling, our proposed framework can efficiently cut down the power consumption without violating the SLA of user requests, whereas [27] treats the datacenter as a *black box*. Third, [27] heavily relies on an initial prediction of entire future workload. However, in the context of a production cloud, it may not be feasible to make accurate predictions of the entire future workload, due to the fact that they are, in general, bursty and nonstationary. One advantage of our framework is that it does not need any future knowledge of workload, and can makes online decisions to enforce the long-term emission budget.

VI. Conclusion

In response to the enormous energy demand and carbon emission of the geographically distributed datacenters, this paper investigates the possibility of exploiting the spatial and temporal variabilities to cut down the carbon emission of the cloud. Specifically, we designed and analyzed a carbon-aware online control framework to balance the three-way tradeoff between electricity cost, SLA requirement and emission reduction budget. By applying Lyapunov optimization techniques, our carbon-aware online control framework can dynamically make decisions on three important control decisions, including geographical load balancing, capacity right-sizing, and server speed scaling. We prove that our carbon-aware online control framework can approach a delicate $[O(1/V), O(V)]$ cost-emission tradeoff. Specifically, the achieved tradeoff allows the geo-distributed cloud to achieve a time-averaged electricity cost that is arbitrarily close to the optimum, while still maintaining the long-term carbon emission budget. Through simulations with empirical real-world workload traces, electricity price and generation data, we demonstrate the effectiveness of our proposed framework in both minimizing electricity cost and cutting down carbon emission.

References

[1] P. X. Gao, A. R. Curtis, B. Wang, and S. Keshav, "It's Not Easy Being Green," in *Proc. of ACM SIGCOMM*, 2012.

[2] Google Green. [Online]. Available: http://www.google.com/green/bigpicture/#/intro/infographics-1

[3] The Guardian. [Online]. Available: http://www.guardian.co.uk/environment/2011/sep/08/google-carbon-footprint

[4] 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.

[5] L. Rao, X. Li, M. D. Ilic, and J. Liu, "Distributed coordination of internet data centers under multiregional electricity markets," *Proceedings of IEEE*, vol. 100, no. 1, pp. 269–282, 2012.

[6] 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. of IEEE INFOCOM*, 2012.

[7] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. H. Andrew, "Greening Geographic Load Balancing," in *Proc. of ACM Sigmetrics*, 2011.

[8] Z. Zhou, F. Liu, H. Jin, B. Li, B. Li, and H. Jiang, "On Arbitrating the Power-Performance Tradeoff in SaaS Clouds," in *Proc. of IEEE INFOCOM*, 2013.

[9] A. Wierman, L. Andrew, and A. Tang, "Power-Aware Speed Scaling in Processor Sharing Systems," in *Proc. of IEEE INFOCOM*, 2009.

[10] Emissions Trading. [Online]. Available: http://en.wikipedia.org/wiki/Emissions_trading

[11] Google Leads Latest Greenpeace Climate Ranking. [Online]. Available: http://www.greenpeace.org/international/

[12] Federal Energy Regulatory Commission. [Online]. Available: https://www.ferc.gov/

[13] Georgiadis, M. J. Neely, and L. Tassiulas, "Resource Allocation and Cross-Layer Control in Wireless Networks," *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1–149, 2006.

[14] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool, 2010.

[15] D. Narayanan, A. Donnelly, and A. Rowstron, "Write off-loading: Pratical power management for enterprise storage," in *Proc. of USENIX Conference on File and Storage Technologies (FAST)*, 2009.

[16] The Green Grid. [Online]. Available: http://www.thegreengrid.org

[17] A. Greenberg, J. Hamilton, D. Malta, and P. Patel, "The Cost of a Cloud: Research Problems in Data Center Networks," in *Proc. of ACM Sigcomm Computer Communication Review*, 2008.

[18] F. Liu, Y. Sun, B. Li, B. Li, and X. Zhang, "FS2You: Peer-Assisted Semi-Persistent Online Hosting at a Large Scale," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 10, pp. 1442–1457, 2010.

[19] ISO Express. [Online]. Available: http://isoexpress.iso-ne.com/guest-hub

[20] P. Shu, F. Liu, H. Jin, M. Chen, F. Wen, Y. Qu, and B. Li, "eTime: Energy-Efficient Transmission between Cloud and Mobile Devices," in *Proc. of IEEE INFOCOM*, 2013.

[21] S. Boyd and L. Vandenberghe, *Convex Optimization*. Combridge University Press, 2004.

[22] J. Guo, F. Liu, D. Zeng, J. C. Lui, and H. Jin, "A Cooperative Game Based Allocation for Sharing Data Center Networks," in *Proc. of IEEE INFOCOM*, 2013.

[23] A. M. Geoffrion, "Generized benders decomposition," *Jouanal of Opyimization Theory and Applications*, vol. 10, no. 4, pp. 237–260, 1972.

[24] M. J. Neely, "Delay-Based Network Utility Maximization," in *Proc. of IEEE INFOCOM*, 2010.

[25] W. Deng, F. Liu, H. Jin, C. Wu, and X. Liu, "MultiGreen: Cost-Minimizing Multi-source Datacenter Power Supply with Online Control," in *Proc. of ACM e-Energy*, 2013.

[26] W. Deng, F. Liu, H. Jin, and C. Wu, "SmartDPSS: Cost-Minimizing Multi-source Power Supply for Datacenters with Arbitrary Demand," in *Proc. of IEEE ICDCS*, 2013.

[27] C. Ren, D. Wang, B. Urgaonkar, and A. Sivasubramaniam, "Carbon-Aware Energy Capacity Planning for Datacenters," in *Proc. of IEEE MASCOTS*, 2012.

APPENDIX A
PROOF OF LEMMA 1

Proof: Squaring Eq. (10) and leveraging the fact that $(\max[a-b+c,0])^2 \leq a^2+b^2+c^2-2a(b-c), \forall a,b,c \geq 0$, we have:

$$\begin{aligned} Q_j^2(t+1)-Q_j^2(t) &\leq C_j^2+E_j^2(t)C_j^2(t) \\ &- 2Q_j(t)[C_j-E_j(t)C_j(t)]. \end{aligned} \quad (19)$$

Based on Eq. (19), we have:

$$\begin{aligned} \Delta(\mathbf{Q}(t)) &\leq \sum_{j=1}^{N}\frac{1}{2}\mathbb{E}\{C_j^2+E_j^2(t)C_j^2(t)|\mathbf{Q}(t)\} \\ &- \sum_{j=1}^{N}Q_j(t)\mathbb{E}\{C_j-E_j(t)C_j(t)|\mathbf{Q}(t)\}. \end{aligned}$$

Note that $E_j(t)C_j(t)$ is bounded by $C_{max} = \max_{j,t} E_j(t)C_j(t)$. By defining $B \triangleq \frac{1}{2}(\sum_{j=1}^{N} C_j^2 + NC_{max}^2)$, the above expression can be simplified to Eq. (14). ■

APPENDIX B
SOLUTION OF PROBLEM (16)

A. Decomposition of Transformed Problem (16)

We first decompose Problem (16) to two sub-problems by defining $R_j(t) = \sum_{i=1}^{M} R_{ij}(t)$, which represents the workload distributed to datacenter j at time slot t. Then, we obtain an assignment problem (20) which determines $R_{ij}(t)$ according to $R_j(t)$ and $A_i(t)$ as follows:

$$\begin{aligned} &\sum_{i=1}^{M} R_{ij}(t) = R_j(t), \\ &\sum_{j=1}^{N} R_{ij}(t) = A_i(t), \\ &R_{ij}(t) \geq 0. \end{aligned} \quad (20)$$

As well as a transformed optimal problem (21) which determines $R_j(t), m_j(t)$ and $\mu_j(t)$ by minimizing the objective (16) as follows

$$\begin{aligned} \min_{R_j(t),m_j(t),\mu_j(t)} \quad & \sum_{j=1}^{N} E_j(t)[VP_j(t)+Q_j(t)C_j(t)], \\ \text{s.t.} \quad & R_j(t) \leq m_j(t)\mu_j(t) - \frac{1}{W_j}, \\ & \sum_{j=1}^{N} R_j(t) = A(t), \\ & \text{and constraints } (3),(4), \end{aligned} \quad (21)$$

where $A(t)$ represents the total workload arriving to the system at time slot t.

The Problem (20) is a simple assignment problem and can be solved by existing algorithms in graph theory. Next, we'll focus on solving Problem (21).

B. Solution of Problem (21)

As in [5], by using the KKT conditions, we can conclude that the optimal solution, if exists, must satisfy the following:

$$R_j(t) = m_j(t)\mu_j(t) - \frac{1}{W_j}.$$

Hence, we can rewrite the optimal problem (21) to a non-linear optimization problem (22) according to the equality constraint in problem (20).

$$\begin{aligned} \min_{R_j(t),m_j(t),\mu_j(t)} \quad & \sum_{j=1}^{N} E_j(t)[VP_j(t)+Q_j(t)C_j(t)], \\ \text{s.t.} \quad & \sum_{j=1}^{N} m_j(t)\mu_j(t) - \frac{1}{W_j} = A(t), \\ & \text{and constraints } (3),(4). \end{aligned} \quad (22)$$

To solve the problem (22), we extend the efficient GBD [5] technique. Some basic definitions should be given here at first.

We define the optimization problem function $F(m,\mu)$ and the constraint function $G(m,\mu)$ as follow:

$$F(m,\mu) = \sum_{j=1}^{N} E_j(t)[VP_j(t)+C_j(t)Q_j(t)],$$

$$G(m,\mu) = A(t) - \sum_{j=1}^{N}[m_j(t)\mu_j(t) - \frac{1}{W_j}].$$

Considering the fact that $V, P_j(t), C_j(t), Q_j(t)$ and PUE_j are already known at time slot t. So function $F(m,\mu)$ can be transformed to

$$F(m,\mu) = \sum_{j=1}^{N} K_j(t)\cdot m_j(t)\cdot[\alpha_j\mu_j^{\nu_j}(t)+\beta_j],$$

where $K_j(t)$ can be treated as a constant at time slot t for $K_j(t) = \text{PUE}_j[VP_j(t)+C_j(t)Q_j(t)]$.

Let $M = \{m_j | m_j \in (0, M_j], \forall j \in \mathcal{D}\}$, $N = \{\mu_j | \mu_j \in (0, s_j], \forall j \in \mathcal{D}\}$ and $I = \{m \in M | G(m,\mu) \leq 0, \mu \in N\}$. For any fixed $\overline{m} \in I$, define the subproblem (23) as follows:

$$\begin{aligned} \min_{\mu \in N} \quad & F(\overline{m},\mu), \\ \text{s.t.} \quad & G(\overline{m},\mu) \leq 0. \end{aligned} \quad (23)$$

And the explicit definition of I:

if $\overline{m} \in M$ and satisfies

$$\inf_{\mu \in N} \Psi^T G(\overline{m},\mu) \leq 0, \forall \Psi \in \Lambda,$$

where $\Lambda \equiv \{\Psi \in R^q | \Psi \geq 0 \text{ and } \sum_{i=1}^{q} \Psi_i = 1\}$,

then $\overline{m} \in I$

Since the constraint function $G(m,\mu)$ is one-dimensional, we can obtain $q=1$. Then we conclude that $\Psi \equiv 1$.

Let $L(m,\lambda) = \inf_{\mu\in S} F(m,\mu) + \lambda G(m,\mu)$, where λ is the optimal multiplier for Problem (23). We define the master problem as follows:

$$\begin{aligned} \min_{m\in M} \quad & \alpha, \\ \text{s.t.} \quad & \alpha \geq L(m,\lambda), \forall \lambda \geq 0, \\ & 0 \geq \inf_{\mu\in S} \Psi^T G(m,\mu), \forall \Psi \in \Lambda. \end{aligned}$$

We have $\Psi = 1$ here according to definition of I.

Based on the above definitions, an iterative schema is developed as follows.

Algorithm 2: Algorithm for Solving Problem (21)

Step 1. Set $p=1$,choose $m^1 \in I$. Solve the subproblem (23), so we obtain an optimal solution μ^1 and an optimal multiplier λ^1. Set $UB = F(m^1,\mu^1)$.
Step 2. Solve the current relaxed master problem (24) by any applicable algorithm.

$$\begin{aligned} \min_{m\in M} \quad & \alpha, \\ \text{s.t.} \quad & \alpha \geq L(m,\lambda^j), j=1,2,...,p, \\ & 0 \geq \inf_{\mu\in S} G(m,\mu). \end{aligned} \tag{24}$$

Let $(\overline{m},\overline{\mu},\overline{\alpha})$ be the optimal solution. Set $LB = \overline{\alpha}$. If $LB \geq UB$,then terminate, and the final optimal solution of the original problem is $(\overline{m},\overline{\mu},\overline{\alpha})$.Else, increase p by 1, and set $m^p = \overline{m}$.
Step 3. Solve the updated subproblem (23). Obtain an optimal solution μ^p and an optimal multiplier λ^p. If $F(m^p,\mu^p) \leq LB$, then terminate, and the final optimal solution is $(m^p,\mu^p,F(m^p,\mu^p))$. Else, if $F(m^p,\mu^p) \leq UB$, set $UB = F(m^p,\mu^p)$. Return to Step 2.

APPENDIX C
PROOF OF THEOREM 1

To prove the time-averaged electricity cost and queue backlog bound in Eq. (17), Eq. (18) in Theorem 1, we have the following Lemma 2:

Lemma 2: (Existence of Optimal Randomized Stationary Policy): There exists at least one randomized stationary control policy π that chooses feasible control decisions $R_{ij}^{\pi}(t), m_j^{\pi}(t), \mu_j^{\pi}(t)$ for $\forall i \in \mathcal{S}, \forall j \in \mathcal{D}, \forall t$, independent of the current queue backlogs, and yields the following steady state values:

$$\begin{aligned} \sum_{j=1}^{N} \mathbb{E}\{E_j(t)P_j(t)\} &= P^*, \\ \mathbb{E}\{E_j(t)C_j(t)\} &\leq C_j, \\ \Rightarrow \mathbb{E}\{E_j(t)C_j(t)\} &\leq C_j - \epsilon, \epsilon \geq 0, \end{aligned} \tag{25}$$

where P^* is defined as the theoretical lower bound on the time-averaged electricity cost. As Lemma 2 can be proved by applying similar techniques as [14], we omit the details for brevity. Based on Lemma 2, we can now to prove the time-averaged electricity cost and queue backlog bound in Eq. (17) and Eq. (18) under our **COCA** algorithm as follows.

Recall that our **COCA** algorithm seeks to choose those decision variables that can minimize the right-hand-side of inequality (15) among all feasible decisions (including the control decision π in Lemma 2) in each time slot, by applying Eq. (25) to inequality (15), we obtain:

$$\begin{aligned} \Delta(\mathbf{Q}(t)) + V\sum_{j=1}^{N} \mathbb{E}\{E_j(t)P_j(t)|\mathbf{Q}(t)\} \leq B \\ -\epsilon\sum_{j=1}^{N} Q_j(t) + VP^*. \end{aligned} \tag{26}$$

Taking an expectation for inequality (26) with respect to the distribution of $Q_j(t)$ and using iterative expectation law results in:

$$\begin{aligned} \mathbb{E}\{L(\mathbf{Q}(t+1)) - L(\mathbf{Q}(t))\} + V\sum_{j=1}^{N} \mathbb{E}\{E_j(t)P_j(t)\} \leq \\ B - \epsilon\sum_{j=1}^{N} \mathbb{E}\{Q_j(t)\} + VP^*. \end{aligned} \tag{27}$$

Summing the above over time slots $t \in \{0,1,...,T-1\}$ and then dividing the result by T, we have:

$$\begin{aligned} \frac{\mathbb{E}\{L(\mathbf{Q}(T))\} - \mathbb{E}\{L(\mathbf{Q}(0))\}}{T} + \frac{V}{T}\sum_{t=0}^{T-1}\sum_{j=1}^{N} \mathbb{E}\{E_j(t)P_j(t)\} \\ \leq B - \frac{\epsilon}{T}\sum_{t=0}^{T-1}\sum_{j=1}^{N} \mathbb{E}\{Q_j(t)\} + VP^* \end{aligned} \tag{28}$$

Rearranging terms and considering the fact that $L(\mathbf{\Theta}(t)) \geq 0$ and $E_j(t)P_j(t) \geq 0$ yields:

$$\frac{1}{T}\sum_{t=0}^{t-1}\sum_{j=1}^{N} \mathbb{E}\{Q_j(t)\} \leq \frac{B + VP^* + \mathbb{E}\{L(\mathbf{Q}(0))\}/T}{\epsilon}$$

Taking limits as $T \to \infty$ results in the time-averaged backlog bound in Eq. (18).

Similarly rearranging inequality (28) we obtain:

$$\frac{1}{T}\sum_{t=0}^{t-1}\sum_{j=1}^{N} \mathbb{E}\{E_j(t)P_j(t)\} \leq P^* + \frac{B}{V} + \frac{\mathbb{E}\{L(\mathbf{Q}(0))\}}{T}$$

Again taking limits as $T \to \infty$ results in Eq. (17).