This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing

Exploiting Spatio-Temporal Diversity for Water Saving in Geo-Distributed Data Centers

Mohammad A. Islam, Kishwar Ahmed, Hong Xu, Nguyen H. Tran, Gang Quan, Shaolei Ren

Abstract—As the critical infrastructure for supporting Internet and cloud computing services, massive geo-distributed data centers are notorious for their huge electricity appetites and carbon footprints. Nonetheless, a lesser-known fact is that data centers are also "thirsty": to operate data centers, millions of gallons of water are required for cooling and electricity production. The existing water-saving techniques primarily focus on improved "engineering" (e.g., upgrading to air economizer cooling, diverting recycled/sea water instead of potable water) and do not apply to all data centers due to high upfront capital costs and/or location restrictions. In this paper, we propose a software-based approach towards water conservation by exploiting the inherent spatio-temporal diversity of water efficiency across geo-distributed data centers. Specifically, we propose a batch job scheduling algorithm, called WACE (minimization of WAter, Carbon and Electricity cost), which dynamically adjusts geographic load balancing and resource provisioning to minimize the water consumption along with carbon emission and electricity cost while satisfying average delay performance requirement. WACE can be implemented online without foreseeing the far future information and yields a total cost (incorporating electricity cost, water consumption and carbon emission) that is provably close to the optimal algorithm with lookahead information. Finally, we validate WACE through a trace-based simulation study and show that WACE outperforms state-of-the-art benchmarks: 25% water saving while incurring an acceptable delay increase. We also extend WACE to joint scheduling of batch workloads and delay-sensitive interactive workloads for further water footprint reduction in geo-distributed data centers.

Keywords—Capacity provisioning, data center, geographic load distribution, resource management, water footprint

1 Introduction

Data centers, housing tens of thousands of servers to satiate the ever increasing demand for Internet and cloud computing services, are notorious for high energy consumption. This raises serious concerns for data centers' operational costs and sustainability impacts due to carbon footprints embedded in electricity usage. While many recent studies have focused on decreasing energy consumption (e.g., [1], [2]) as well as carbon footprint of data centers (e.g., [3], [4]) for sustainability, an equally, if not more, important yet often neglected aspect of data center sustainability is the massive water footprint.

1.1 Why data center water footprint matters?

Massive water footprint. Data centers consume a significant amount of water both directly and indirectly. Direct onsite water consumption is attributed to cooling systems. While there exist various types of cooling systems such as air-side economizer [5], large data centers, including AT&T [6] and eBay [7], often resort to cooling towers, where server heat is rejected into the environment via water evaporation. It is reported that the U.S. National Security Agency's data center in Utah would require up to 1.7 million gallons of water per day, enough to satisfy over 10,000 household's water needs. Cooling facilities of eBay consume 2.52 liters of water per kilowatt-hour (L/kWh) server energy, as of 2013 summer [7]. Even outside air cooling, which is particularly suitable for cold climate, consumes water for temperature/humidity control (e.g., Facebook's data center uses 0.42 L/kWh as of February 2014 [8]). In addition to direct onsite water consumption, data centers are also held responsible for an enormous amount of *indirect* water embedded in electricity generation: electricity production accounts for the largest water withdrawal in the U.S. and an average of 1.8L of water is evaporated for just 1KWh of electricity generation (even excluding the much more water-consuming hydropower) [9], [10].

Extended droughts. Extended droughts and water shortage are quickly spreading as a global crisis, amid the anticipation that global water demand may exceed the supply by 40% in 2030 [11], [12]. Even in the U.S., over 70% of the land area was affected by drought during 2012 [13]. The situation has become even worse in 2014 for some of the U.S. states: following a three-year abnormal dryness, California declared *drought emergency* on January 17, 2014, urging its residents to cut water usage by at least 20% [14]. Even in water-abundant regions, water conservation can benefit data centers in

- *This work was supported in part by the U.S. NSF under grants CNS-1551661 (CAREER) and CNS-1565474, HK RGC under grant ECS 21201714, and Korea MSIP (under the G-ITRC support program supervised by the IITP) IITP-2015-R6812-15-0001.*
- *Mohammad A. Islam and Shaolei Ren are with University of California, Riverside. Email: misla006@ucr.edu and sren@ece.ucr.edu*
- *Kishwar Ahmed and Gang Quan are with Florida International University. E-mail: kahme006@fiu.edu and gang.quan@fiu.edu*
- *Hong Xu is with City University of Hong Kong. E-mail: henry.xu@cityu.edu.hk*
- *Nguyen H. Tran is with Kyung Hee University. E-mail: nguyenth@khu.ac.kr*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing

acquiring green certifications (such as LEED program, which 77% large data centers are seeking, as shown in a recent survey [15]), tax credits [16], government-mandated water compliance code [17], [18], fulfilling corporate social responsibility and mitigating business continuity [6].

1.2 How to conserve water in data centers?

Despite the emergence of water footprint as a critical concern, little prior research has been done to address this issue. Some large IT companies such as Google and Facebook have recently become concerned about the tremendous amount of water usage by their data centers, and are developing new techniques to mitigate the water consumption (e.g., applying air economizer instead of cooling towers in cold regions, using recycled/sea water instead of potable water [8], [19]). However, these engineering approaches often require high upfront costs and/or suitable locations/climates, which significantly limit their applicability to all data centers and hence necessitate a more "universal" approach that will be addressed in this paper.

Although water conservation and energy saving are related, most of the current research on data center energy optimization (e.g., [4], [20], [21]) is insufficient for water conservation. This is because existing studies do not consider *spatio-temporal* diversity of data center water efficiency: 1) temporal diversity results from volatile weather condition and time-varying energy fuel mixes of electricity generation; and 2) spatial variation is because different data center locations (i.e., states) constitute varying amount of energy fuel mix (details are provided in Section 3). Moreover, water efficiency is not equivalent to electricity cost/carbon efficiency (e.g., nuclear power consumes more than 2L of water per kWh, while incurring little carbon emission).

Some recent studies [22], [23] exploit *spatial* diversity of water efficiency by scheduling delay-sensitive interactive workloads among data centers to reduce water footprint. However, they neglect the temporal diversity of water efficiency, which allows untapped water saving opportunities through dynamically scheduling delay-tolerant jobs over time.

Our work. We address the dearth as well as urgency of data center water conservation by integrating spatio-temporal diversity of water efficiency into workload scheduling and resource provisioning decisions for geo-distributed data centers. Our approach judiciously decides "*when*" and "*where*" to process workloads as well as how much computing resource needs to be provisioned, based on three widely-available control knobs: workload scheduling (both across and within data centers), turning on/off servers, and adjusting server processing speeds (via dynamic voltage and frequency scaling, or DVFS). While these three have been extensively studied in various contexts ([1], [4], [24]), the *uniqueness* of our research is that we integrate spatio-temporal diversity of water efficiency and propose a new scheduling algorithm to tune the knobs for water conservation.

It is a challenging problem to reduce water footprint via resource management. Intuitively, we would like to process more jobs in data centers with higher water efficiency and/or when water efficiency is higher. Nonetheless, naive techniques will result in two undesirable consequences: (1) electricity cost may be significantly compromised, and (2) jobs will experience an intolerable delay if they are only processed in very water-efficient times, whereas water may be unnecessarily wasted if temporal diversity of water efficiency is neglected. Moreover, the time-varying nature of water efficiency (resulting from volatile outside temperature and energy fuel mixes), job arrival, carbon emission rate and electricity price adds further challenges to making resource management decisions over a long timescale, since it is difficult to accurately foresee "when" water efficiency is high.

To address the above challenges, we focus on delay-tolerant batch jobs and propose a provably-efficient online batch job scheduling algorithm, called WACE (minimization of WAter, Carbon and Electricity cost), to minimize water usage while also considering electricity cost, carbon emission and delay performance in geo-distributed data centers. WACE exploits the spatio-temporal variation in water efficiency, electricity price as well as carbon emission rate, and dynamically dispatches workloads to data centers while satisfying delay performance requirement. We conduct a trace-based simulation to validate WACE. The results show that: (1) WACE can reduce the total cost while satisfying the average delay constraint; and (2) compared with state-of-the-art scheduling algorithms, WACE can reduce the total cost by approximately 20%, while reducing the water consumption by approximately 25%. Finally, we extend WACE to jointly schedule delay-tolerant batch jobs and delay-sensitive interactive jobs for further water footprint reduction in geo-distributed data centers.

To sum up, we take the position that addressing the enormous water footprint is essential for data center operation, especially in water-stressed regions. We make the following contributions. First, as compared to the existing research that jointly considers workload latency, electricity cost and carbon footprint [4], we reduce data centers' water footprint by incorporating water conservation as a complementary yet critical criterion into resource management, which further extends the current scope of research on data center sustainability. Second, we propose a provably-efficient batch job scheduling algorithm, WACE, which can be implemented online without foreseeing the far future information. Third, we evaluate WACE and demonstrate its effectiveness using simulations based on real-world traces. Last but not least, we consider jointly scheduling delay-tolerant batch jobs and delay-sensitive interactive jobs for water conservation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing

TABLE 1: List of key notations.

Notation	Description
$a(t)$	Batch workload arrival
$m_i(t)$	# of active servers at data center i
$s_i(t)$	Server processing speed at data center i
$p_i(t)$	Server energy consumption at data center i
$w_i(t)$	Water consumption at data center i
$c_i(t)$	Carbon emission at data center i
$e_i(t)$	Electricity cost at data center i
$g(t)$	Total cost
V	Cost-delay parameter
$J_i(t)$	Job queue for each data center i

2 MODEL

In this section, we lay down the system model for our resource management-based approach towards water conservation. We consider a discrete-time framework to model time-varying factors. The entire time horizon of interest is divided into K time slots of equal length (e.g., in the order of minutes or one hour), which we denote by $t = 0, 1, \cdots, K-1$. Resource management decisions are made at the beginning of each decision time slot. Key notations are summarized in Table 1. Next, we present the modelling details for data centers and workloads that capture three widely-available control knobs: workload scheduling (both across and within data centers), turning on/off servers, and adjusting server processing speeds (via DVFS).

2.1 Workload

In general, there are two types of workloads in data centers: batch workloads and interactive workloads. Many batch jobs are delay-tolerant and usually do not require to be processed immediately after arrival, as long as they do not get stalled unreasonably long. Examples of such jobs are Google's search indexing, periodical data backups, scientific computing, etc. Therefore, compared to interactive jobs (e.g., web services or business transactional applications) that typically have a delay requirement in the order of tens of milliseconds, a larger delay is acceptable for batch job processing, allowing the data center operator to leverage the temporal variation of water-carbon efficiency and electricity price. We will first focus on scheduling batch jobs, while extension of jointly scheduling batch and interactive jobs is available in Section 5. We denote by $a(t)$ the total amount of batch job arrivals at time t, while the amount of batch jobs dispatched to data center i is $a_i(t)$ subject to $a(t) = \sum_{i=1}^{N} a_i(t)$. We measure the batch workloads in terms of machine-time, as considered widely in existing literature [25], [26] and can successfully guide dynamic provisioning of computing resources. Note that our model can also be extended to include the additional constraint that certain types of jobs can only be dispatched to a subset of data centers for processing due to, for example, data availability constraints.[1]

2.2 Data center

We consider N geo-distributed data centers denoted by $i \in \{1, 2, \cdots, N\}$. Each data center has an auxiliary onsite renewable energy source (e.g., solar panels [27]), and we denote the amount of available renewable energy at time t by $r_i(t)$ for data center i. There are a total of $M_i(t)$ servers that are homogeneous and available for processing batch jobs within data center i at time t. Note that our model is easily extensible to heterogeneous servers and, if so, both number and type of servers allocated to process jobs need to be decided. Servers may run at different processing speeds and incur different power via DVFS [28]. Specifically, we consider an array of finite processing speeds denoted by $\mathcal{S}_i = \{s_{i,1}, \cdots, s_{i,K_i}\}$, from which a speed s_i is chosen for processing batch workloads for data center i. In general, server power consumption is related to a variety of resources, including CPU, memory and disk. However, since CPU is regarded as the main contributor to power consumption of a server (besides idle power), we focus on CPU utilization, while suppressing other components when calculating server power consumption [1], [29]. Hence, we express the total power consumption of a server in data center i at time t as $\alpha_i \cdot s_i^{n_i}(t) + p_{0,i}$ [30], [31], where α_i is a positive scaling factor and relates the processing speed to the power consumption; $p_{0,i}$ represents the power consumption in idle or static state; and the exponent parameter n_i is determined through empirical methods (example values of which range from 1 to 3, and can be found in [28], [30], [32]).

Based on the above power model, we now derive data center power consumption at time t, which is also equivalent to energy consumption in our model because all time slots have the same duration and hence power and energy is used interchangeably. For convenience of presentation, we simply model the server energy consumption by interactive workloads in data center i as an exogenously-determined value $p_{i,int}(t)$. However, we will include interactive workloads in Section 5 and show further improvement of such inclusion. Letting $s_i(t)$ and $m_i(t)$ be processing speed and the number of active servers for processing batch workloads, respectively, we can write the total server energy consumption at data center i as

$$\begin{aligned} p_i(t) &= p_{i,bat}(t) + p_{i,int}(t) \\ &= m_i(t) \cdot [\alpha_i \cdot s_i^{n_i}(t) + p_0] + p_{i,int}(t). \end{aligned} \tag{1}$$

Note that we do not consider server utilization to determine $p_{i,bat}(t)$ as batch jobs can be equivalently considered as running at a constantly high utilization. The reason is batch jobs are processed at servers' maximum

1. As specified by the queueing dynamics (in Section 3.3), batch job arrivals at time t will not be available for processing until time $t+1$, thereby implicitly capturing the possible delay incurred during the load dispatching stage (e.g., due to data movement).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing

processing capacities given a certain speed, and results in a continuously high server utilization.

Next, given the available on-site renewable energy $r_i(t)$, data center i's electricity usage at time t is $[\gamma_i(t)p_i(t) - r_i(t)]^+$, where $[\ \cdot\ ]^+ = \max\{\cdot, 0\}$ and $\gamma_i(t)$ is the factor of Power Usage Effectiveness (PUE) at data center i capturing the non-IT energy consumption such as cooling and power supply system. Note that, as our focus is on workload scheduling, we do not consider facility management such as battery charging/discharging, which nonetheless can be integrated using other orthogonal techniques [33].

3 Online Batch Job Scheduling: WACE

In this section, we present our online batch job scheduling algorithm WACE. We will first formulate three types of "costs" (e.g., water consumption, electricity cost, and carbon emission), present the problem formulation, and then develop WACE to minimize the total weighted cost in the absence of long-term future information (e.g., future water efficiency and energy fuel mixes, workload arrivals, etc.).

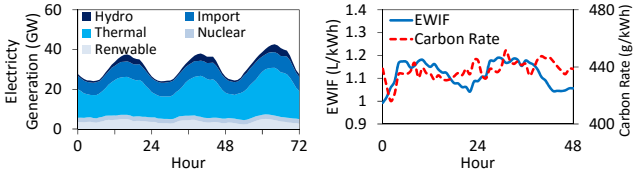
3.1 Cost

Our work aims to address three "costs": water consumption, electricity cost and carbon emission. While electricity cost can be directly represented and quantified in terms of monetary values, water consumption and carbon emission symbolize data center sustainability and can be converted to monetary costs when we present the problem formulation [4].

• **Water consumption.** Data centers consume a significant amount of water both "directly" (for cooling) and "indirectly" (for electricity generation) [10], [37]. In what follows, we provide a brief sketch of these two types of water consumption, introduce the notion of Water Usage Effectiveness (or WUE, which measures water efficiency), and then formulate water consumption in a data center.

Direct water: Data centers deploy one or more of several heat removal methods, among which cooling towers are widely used in large data centers [5], [10]. Cooling tower consists of two water loops: chilled water loop and condenser water loop. While chilled water loop recirculates between chillers and server rooms, condenser water evaporates and recirculates between chillers and cooling towers. Cooling towers consume water in two major ways: evaporation (rejection of heat into the environment) and "blown down" (to keep the salt concentration of condenser water at a low level). Given a cooling tower, direct WUE at time t at data center i is denoted by $\epsilon_{i,D}(t)$. In general, direct WUE can be directly measured in realtime, and some data centers have been periodically reporting direct WUE for public access (e.g., dashboard of Facebook, eBay [7], [8]).

Indirect water: Electricity production accounts for the largest water withdrawal among all sectors in the U.S. [38], [39]. While not all the water withdrawal evaporates/loses (hence considered as "consumed"), and a small portion of electricity is produced through water-free renewable sources (e.g., wind, solar PV), the national average water consumption for electricity production in the U.S. still reaches 1.8L/kWh, even without considering hydroelectricity[2] which itself is a huge water consumer [10]. Water efficiency for electricity production is quantified in terms of Energy Water Intensity Factor (EWIF), which measures the amount of water consumption per kWh electricity. Table 2 shows the EWIF of several common energy fuel mixes.



(a) 3-day fuel mix in California starting from Sep 1, 2013.

(b) EWIF and carbon emission rate in California.

Fig. 1: Fuel mix and water/carbon efficiency [40].

Based on [4], we use the following formula to calculate the grid power's water efficiency:

$$\epsilon_{i,I}(t) = \frac{\sum_k q_{i,k}(t) \times \epsilon_{i,k}}{\sum_k q_{i,k}(t)} \tag{2}$$

where $q_{i,k}(t)$ denotes the amount of electricity generated and $\epsilon_{i,k}$ denotes EWIF for fuel type k in data center i.

Water usage efficiency: To assess the water usage efficiency for data center operation, an emerging metric, called WUE, was recently developed by The Green Grid [10]. At a high level, WUE is the ratio of water consumption to IT equipment energy, where water consumption includes both direct and indirect water consumption (i.e., water evaporated/"lost" for on-site data center cooling and off-site electricity production at power plants [39]).

Now, we formulate the water consumption at time t for data center i as follows

$$w_i(t) = \epsilon_{i,D}(t) \cdot p_i(t) + \epsilon_{i,I}(t) \cdot [\gamma_i(t) \cdot p_i(t) - r_i(t)]^+, \tag{3}$$

where $p_i(t)$ is the server power, $\epsilon_{i,D}(t)$ denotes the direct on-site WUE for cooling system, $\epsilon_{i,I}(t)$ is the EWIF calculated based on (2), $\gamma_i(t)$ is the PUE.

Spatio-temporal diversity of WUE: WUE (both direct and indirect) shows spatial and temporal variation. For example, indirect WUE is location-specific [10]: different states in the U.S. demonstrates significant variation in EWIF because of different energy fuel mixes and/or cooling systems at power plants. Direct WUE also varies by location, as corroborated by observing direct WUE values at two data centers of Facebook [8]. Temporal variation of EWIF can be seen in Fig. 1(b): it is evident

2. Hydroelectricity is often excluded in the assessment of grid's water efficiency, because water is mostly consumed due to indirectly expedited surface water evaporation during its generation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing

TABLE 2: EWIF and carbon emission rate of common electricity generation methods [34]–[36].

Fuel Type	Renewable	Nuclear	Thermal	Imports	Hydro
EWIF (L/kWh)	0.225	2.27	1.13	1.8	78.9 (or 0 if excluded)
Carbon (g/kWh)	22.5	15	766	562	13.5

that energy fuel mix is time-varying and hence different amounts of water are consumed at different times for the same amount of electricity. Direct WUE is also time-varying because of non-stationary outside temperatures for cooling tower locations [5].

• **Electricity cost.** We denote the electricity price for data center i at time t by $u_i(t)$ which may change over time/locations. We can express the incurred electricity cost at data center i during time t as:

$$e_i(t) = u_i(t) \cdot [\gamma_i(t) \cdot p_i(t) - r_i(t)]^+, \tag{4}$$

where $\gamma_i(t)$ denotes the PUE of data center i.

• **Carbon emission.** A data center incurs carbon emission embedded in electricity production [4]. Like EWIF, the average carbon emission rate of data center i can be calculated based on the weighted contribution from each fuel type used in electricity generation of the grid [4]. Table 2 lists the carbon emission rates of several common energy fuel mixes.

Now, we express the carbon footprint (ignoring negligible carbon emission from on-site renewable energy generation) of the data center i at time t as

$$c_i(t) = \phi_i(t) \cdot [\gamma_i \cdot p_i(t) - r_i(t)]^+, \tag{5}$$

where $\phi_i(t)$ is the carbon emission rate [4] converting electricity usage to carbon emission and has a unit of g/kWh.

3.2 Problem formulation

In this subsection, the optimization objective and constraints are first specified, and then the problem formulation for online batch job scheduling is presented.

Objective. The optimization objective is to minimize total (operational) cost: electricity cost, water consumption and carbon emission, while ignoring capital costs (e.g., cost for building data centers, installing renewal generators and so on) that are orthogonal to our study. As shown in Fig. 1(b), carbon emission rate (similarly, electricity cost efficiency, albeit not shown) and water efficiency are not aligned: low-carbon electricity may not be water-efficient. Thus, there exists an inherent tradeoff among electricity cost, carbon emission and water consumption. In other words, these three metrics cannot be optimized simultaneously, and hence as in the existing literature [1], [4], we construct a parameterized *total cost* function as follows

$$g(\mathbf{a}(t), \mathbf{m}(t), \mathbf{s}(t)) = \sum_{i=1}^{N} \left[e_i(t) + h_w \cdot w_i(t) + h_c \cdot c_i(t) \right], \tag{6}$$

where $\mathbf{a}(t) = (a_1(t), \cdots, a_N(t))$, $\mathbf{m}(t) = (m_1(t), \cdots, m_N(t))$, and $\mathbf{s}(t) = (s_1(t), \cdots, s_N(t))$ are the decision variables representing geographic workload distribution, number of servers, and server speed settings, respectively. Moreover, $h_w \geq 0$ and $h_c \geq 0$ are weighting parameters for water consumption and carbon emission relative to the electricity cost. Such a multi-objective formulation is common in the literature (e.g., [4] combines electricity cost, carbon emission and delay). Our optimization objective is to minimize the long-term average cost expressed as $\bar{g} = \frac{1}{K} \sum_{t=0}^{K-1} g(t)$, where K is the total number of time slots in the period of interest.

Constraints. We list the constraints on scheduling decisions as follows. First, at any time slot t, the number of available servers to process the batch jobs needs to satisfy

$$0 \leq m_i(t) \leq M_i(t), \tag{7}$$

where $M_i(t)$ is the total number of servers excluding those allocated to interactive workloads. The server can only select one of the supported speeds:

$$s_i(t) \in \mathcal{S}_i = \{s_{i,0}, s_{i,1}, \cdots, s_{i,K_i}\}. \tag{8}$$

We also need to guarantee that batch jobs will be eventually processed (without dropping):

$$a(t) = \sum_{i=1}^{N} a_i(t), \ \forall\, t, \tag{9}$$

$$\bar{a}_i < \bar{b}_i, \ \forall\, i \tag{10}$$

$$b_i(t) = m_i(t) \cdot s_i(t), \forall\, i, t, \tag{11}$$

where $\bar{a}_i = \frac{1}{K} \sum_{t=0}^{K-1} a_i(t)$ and $\bar{b}_i = \frac{1}{K} \sum_{t=1}^{K-1} b_i(t)$ are the long-term average workload arrival and allocated server capacity in data center i, respectively, and the constraint (11) states the relation between the processed batch jobs and data center capacity provisioning. Note that although we do not explicitly incorporate the average delay into the constraints, WACE provides an upper bound on the maximum queue length (details available at [54] which is omitted for brevity), translating into an average queueing delay guarantee.

Problem formulation. We present an offline problem formulation for batch job scheduling as follows

$$\mathbf{P1}: \quad \min_{\mathcal{D}} \bar{g} = \frac{1}{K} \sum_{t=0}^{K-1} g(\mathbf{a}(t), \mathbf{m}(t), \mathbf{s}(t)) \tag{12}$$

$$s.t., \quad \text{constraints } (7), (8), (9), (10), (11), \tag{13}$$

where $\mathcal{D}$ represents a sequence of decisions, i.e., $\mathbf{a}(t), \mathbf{m}(t), \mathbf{s}(t)$, for $t = 0, 1, \cdots, K-1$, which we need to optimize. Clearly, the optimal offline solution to **P1**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing

Algorithm 1 WACE

1: At the beginning of each time t, observe the data center state information $r_i(t)$, $\epsilon_{i,D}(t)$, $\epsilon_{i,I}(t)$, $\phi_i(t)$ and $u_i(t)$, for $i = 1, 2, \cdots, N$ and $t = 0, 1, 2, \cdots, K-1$

2: Choose $\mathbf{a}(t), \mathbf{m}(t), \mathbf{s}(t)$ subject to (7), (8), (9), (11) to minimize

$$\mathbf{P2}: V \cdot g(\mathbf{a}(t), \mathbf{m}(t), \mathbf{s}(t)) - \sum_{i=1}^{N} J_i(t) \cdot b_i(t) + \sum_{i=1}^{N} J_i(t) \cdot a_i(t) \quad (14)$$

3: Update the batch job queue $J_i(t)$ according to (15).

provides the lowest average cost. However, such a solution requires complete offline information (i.e., workload arrivals, direct WUE, EWIF, carbon emission rate, on-site renewables and electricity prices) throughout the entire time horizon, which is challenging, if not impossible, to obtain in practice. Therefore, we resort to an online algorithm that is implementable based on the currently available information at the beginning of each time t.

3.3 WACE

We now present an online batch job scheduling algorithm, WACE, which is proved to yield a close-to-minimum cost compared to the optimal algorithm with lookahead information. To save space, the performance analysis of WACE is presented in [54] which follows sample-path Lyapunov optimization [41].

Batch job decisions are linked together across different time slots through the long-term constraint (10). Thus, it is challenging to make online decisions without knowing the future. Here, we propose an online algorithm that leverages the recently-developed Lyapunov technique [41]. Specifically, we *replace* the long-term constraint (10) with a batch job queue, and incorporate the queue information into scheduling decisions. Intuitively, with a larger queue length, more resources should be allocated to the data center (e.g., increasing number of server and/or server speed), and vice versa. As described in Algorithm 1, we incorporate the queue length information into the objective function to dynamically determine the balance between clearing queue backlogs (i.e., delay) and cost minimization. With an initial empty queue $J_i(0) = 0$, the job queue is updated as follows

$$J_i(t+1) = [J_i(t) - b_i(t)]^+ + a_i(t), \quad (15)$$

where $[\cdot]^+ = \max\{\cdot, 0\}$, $a_i(t)$ quantifies the batch job arrivals, and $b_i(t)$ indicates the amount of processed batch jobs. In Algorithm 1, we use a control parameter $V \geq 0$ (denoted as cost-delay parameter), which can be tuned to trade the batch job queueing delay for cost i.e., how much we shall emphasize the cost minimization problem **P1**, compared to the long-term delay performance. In particular, when the value of V is smaller, the data center operator follows (10) more closely, therefore resulting in higher cost but a lower delay. On the other hand, with larger value of V, data center operator focuses more on minimizing the cost, suppressing the inclusion of the queue length in **P2**. We will further examine the effect of V in the simulation. Note that, in Algorithm 1, the load balancing decision $\mathbf{a}(t)$ is following the rule of "joining the shortest queue" to balance the queue length (and hence queueing delay) across data centers: WACE routes incoming batch jobs to the data center with the shortest queue length, thereby prevents the data center with high job queue from increasing further.

TABLE 3: Data center configuration.

DC Location	# of Servers (in thousands)	Peak Power (MW)
CA	100	21
IA	90	19
IL	60	13
NY	70	15

TABLE 4: Average direct WUE and EWIF [10].

DC Location	CA	IA	IL	NY
EWIF (L/KWh)	0.19	0.45	3.97	3.22
Direct WUE (L/KWh)	0.7	2.52	0.2	0.49

4 Performance Evaluation

In this section, we present trace based simulation studies of geo-distributed data centers to validate our analysis and evaluate the performance of WACE. First, we present the data used, and then we present the simulation results.

4.1 Data sets

We consider four geo-distributed data centers located at: Mountain View (CA), Council Bluffs (IA), Northlake (IL), and New York (NY). The number of servers and peak power of each data center are given in Table 3. By default, PUE of each data center is set to 1.2. Although we have chosen the continental U.S. for our study due to the availability of energy fuel mix and electricity price information, our study is generalizable to the globe, too. Each server has 15 discrete speed levels, uniformly ranging from 1.6 GHz to 3 GHz, and the normalized service rate ranges from 5.3 to 10 jobs per hour, where each "job" represents a unit of computing workloads. We model the power consumption of the servers according to (1). The default weighting parameters for water consumption and carbon emission in (6) are set to $h_w = 25$ and $h_c = 0.06$ respectively to have water and carbon cost comparable to electricity cost. The default average delay requirement for batch jobs is 5 hours. The total simulation period is 1 year and duration of each time slot is set to 1 hour.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing

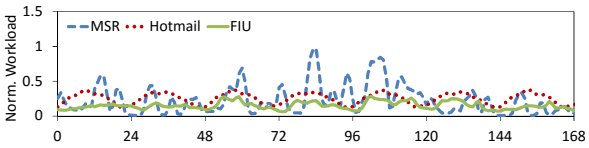


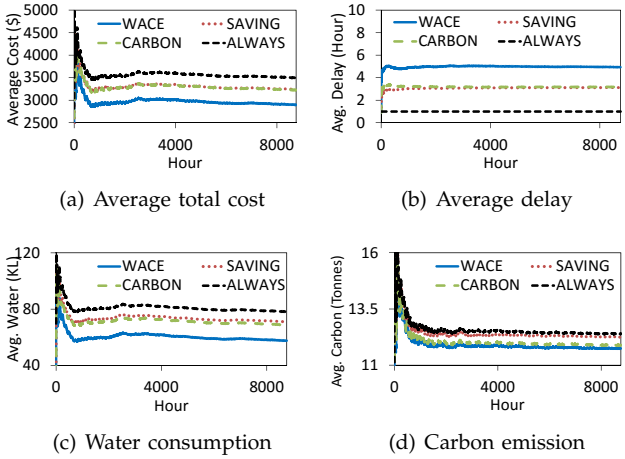
Fig. 2: Workload traces [42].

• **Workload trace:** The batch workload traces for our simulations are taken from the research publication [42] and originally profiled from two real-world data center traces: Microsoft Research (MSR, which we will use as the default batch workload) and Hotmail (used for robustness study). Due to the lack of yearly traces, we repeat both these traces by adding 30% random noises, extend them to one year, and scale them for our data center setting. On average, the default amount of batch workloads is 25% of the total maximum capacity. We profile the HTTP server usage in Florida International University throughout 2012 and use the scaled-up trace as our interactive workload, with an average utilization at 15% of the total maximum capacity. By default, the total interactive workload is distributed to data centers in proportion to their maximum server capacities. Fig. 2 illustrates a snapshot of the MSR, Hotmail and FIU workload traces, where the workloads are normalized with respect to the total maximum capacity of the four data centers.

• **Electricity price and renewable energy:** We obtain the hourly electricity price for three trading nodes closest to data centers in CA, IA and NY for the year of 2012, while we collect electricity prices for the data center in IL from [43]. We obtain from [40] four sets of the hourly renewable energies (generated through solar panels and wind turbines) during the year of 2012, and scale them proportionally such that on-site renewable energy supply takes up approximately 10% of the peak power consumption of all data centers.

• **Others:** We use the data presented in [8] to calculate the direct water consumption in CA data center. For data centers at IA, IL and NY, we use the average direct WUE of 2.52 L/kWh, 0.2 L/kWh and 0.49 L/kWh, respectively, and vary them according to outside wet bulb temperature to incorporate temporal diversity. We use the state-level average EWIF (listed in Table 4) and vary it according to energy fuel mix data collected from [40]. A sample 48 hour energy fuel mix data is given in Fig. 1(a). Due to the unavailability of hourly fuel mix data of IA, IL and NY, we adopt the approach in [4] and use the monthly fuel mix data to calculate monthly hourly EWIF and carbon emission rates, and add 15% randomness to generate the yearly data for these three data centers.

Due to unavailability of access to data center information, we collect the trace-data from various sources. However, since our collected data appropriately represents the variation of workloads, electricity price, water efficiency and carbon rate, our purpose of evaluating WACE is served.



(a) Average total cost (b) Average delay

(c) Water consumption (d) Carbon emission

Fig. 3: Comparison of different algorithms.

4.2 Results

We drive the simulation using the above trace data. The water consumption, carbon emission and electricity cost are recorded as outputs of the simulation. We compare the performance of WACE algorithm against three benchmark algorithms i.e., SAVING, CARBON and ALWAYS. We also present insights on how WACE appropriately captures spatio-temporal variation to outperform the benchmark algorithms.

Next, we provide a brief outline of the algorithms that we compare WACE with.

• **SAVING:** We consider here an online algorithm that solely minimizes the electricity cost while discarding water consumption and carbon emission. Essentially, SAVING is a variant of WACE by setting the water and carbon weights to zero.

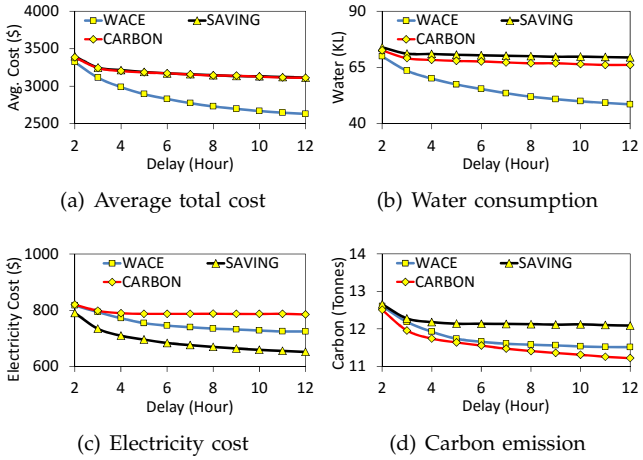
• **CARBON:** CARBON only optimizes the carbon emission while ignoring electricity cost and water consumption. It applies WACE with an "infinite" weight for carbon footprint.

• **ALWAYS:** This algorithm processes the workloads as soon as possible and hence avoids incurring a large delay. By default, ALWAYS equally distributes the workloads to data centers.

4.2.1 *Performance comparison*

Fig. 3 shows the comparison between WACE and three benchmark algorithms, i.e., ALWAYS, SAVING and CARBON, in terms of the average cost, average delay, average water consumption and average carbon emission per time slot. In Fig. 3, the average value is obtained by summing up all the values from time 0 to time t and then dividing the sum by $t+1$. Also, in our study the average hourly cost, water consumption and carbon emission represent the sum for all the data centers, while average queuing delay is averaged over

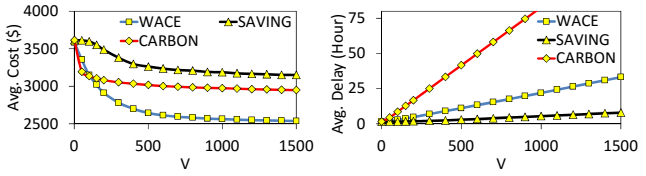
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing



(a) Average total cost (b) Water consumption

(c) Electricity cost (d) Carbon emission

Fig. 4: Impact of average delay on different costs.



(a) Average total cost (b) With increase in V average delay increases

Fig. 5: Impact of cost-delay parameter V.

all the data centers. For WACE, SAVING and CARBON, we choose V to be 210, 55, and 580, respectively. Fig. 3(a) demonstrates that WACE is more cost-effective compared to the three benchmarks with a total cost saving by more than 20%, 11%, and 10%, respectively, although it can be seen from Fig. 3(b) that the queuing delay is increased by approximately 1 hour for WACE compared to benchmarks, which is typically tolerable for batch jobs. Due to its greedy nature, ALWAYS has the lowest delay of 1 hour, but it has the highest cost. From Figs. 3(a) and 3(b), it is evident that WACE takes better advantage of delay tolerance of batch jobs and is therefore capable of achieving lower average total cost by processing batch jobs during time slots when combined cost factor is lower (i.e., incorporating electricity price, water efficiency and carbon emission rate). Figs. 3(c) and 3(d) compare water consumption and carbon emission, respectively, between WACE and three benchmark algorithms. It demonstrates the benefits of WACE in terms of sustainability, while incurring a *minor* delay performance degradation. Note that WACE achieves lower carbon emission than CARBON in Fig. 3(d) because of the choice of control variable V. In particular, CARBON has an average delay of three hours while batch jobs experience an average delay of five hours in WACE. Thus, WACE can better take advantage of the spatio-temporal diversity of carbon/water efficiency and even result in a lower carbon footprint than CARBON.

4.2.2 Impact of average delay

To enable a fairer comparison, we show the performance of WACE, SAVING and CARBON under the same average delay. ALWAYS is not shown here because it does not provide the flexibility of trading delay for cost saving. We vary the delay constraint from 2 time slots to 12 time slots to show the corresponding average cost, electricity cost, water consumption and carbon emission in Fig. 4. We can see from Fig. 4(a) that the average total cost decreases for all the algorithms with relaxed delay constraint, with WACE achieving the lowest average total cost among the three algorithms. With increased delay constraint, WACE performs significantly better than other two algorithms. Moreover, note that increasing the delay beyond 10 hours does not considerably improve the cost performance for any of the algorithms. Fig. 4(b) shows similar pattern in terms of water consumption. We can observe that WACE achieves lowest water consumption among the three algorithms since it considers water optimization, while the other two algorithms neglect water efficiency (which differs from electricity cost/carbon efficiency). Fig. 4(c) and Fig. 4(d) deliver similar messages in terms of electricity cost and carbon emission, respectively, but with SAVING achieving lowest electricity cost and CARBON having lowest carbon emission, which are expected from the operating principle of these algorithms. From Fig. 4, we observe that it is not possible to simultaneously optimize electricity cost, water consumption and carbon emission, because the efficiencies of these three metrics are not strongly correlated. A similar result was reported by [4]: electricity cost and carbon footprint cannot be minimized at the same time. Hence, it is important to properly choose weighting parameters for water and carbon footprints, such that neither sustainability nor economic benefit is considerably compromised.

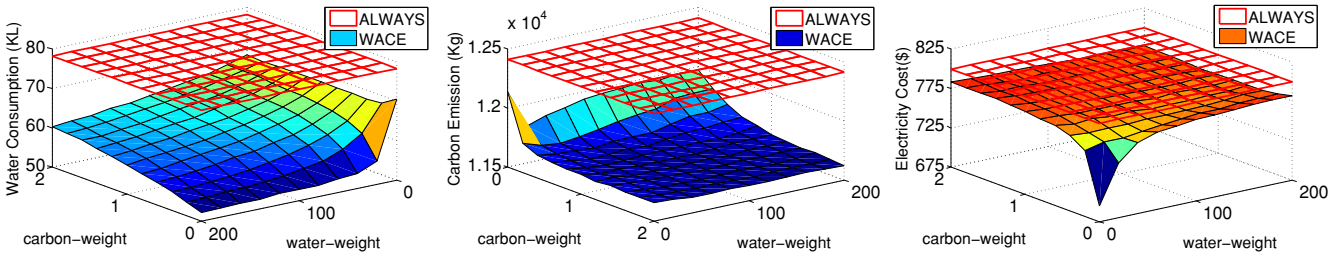
4.2.3 Impact of cost-delay parameter V

We now show how the cost-delay parameter V affects cost and delay. Fig. 5(a) and Fig. 5(b) show the impact of V on the average hourly cost and average queueing delay, respectively. ALWAYS is not considered here, since it does not use V in its scheduling decision. Fig. 5(a) shows that for all the algorithms, with the increase in V, the average cost decreases. In Fig. 5(b), we see that delay increases with an increase in V and the delay increase is almost linear. The result conforms with our analysis that with a greater V, the algorithms are less concerned with the batch job queue length while caring more about minimizing the cost. The reason is that, with a large value of V, the weight of queue length in the optimization objective (14) is relatively smaller, thereby equivalently making the queueing delay less stringent.

4.2.4 Impact of water and carbon weights

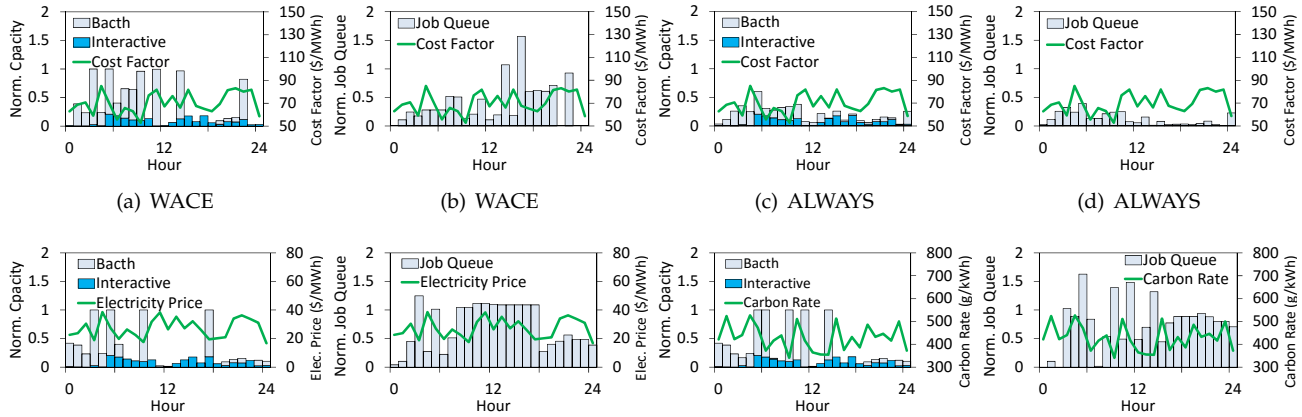
In this section, we show the effect of water weight (h_w) and carbon weight (h_c) on the performance of WACE.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing



(a) Water consumption (b) Carbon emission (c) Electricity cost

Fig. 6: Impact of water and carbon weights.



(a) WACE (b) WACE (c) ALWAYS (d) ALWAYS

(e) SAVING (f) SAVING (g) CARBON (h) CARBON

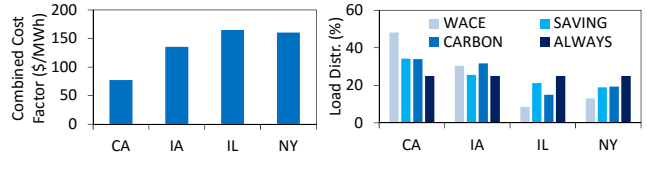
Fig. 7: Capacity provisioning in CA.

We start with the value zero for both the weight factors and increase the values up to the level such that the water/carbon cost equal to 90% of the total cost. We choose V appropriately in each case, such that the average delay is equal to 5 hours. In Fig. 6, we show a set of figures depicting the impacts of water and carbon weights on WACE in terms of different costs. As we have shown that WACE cannot outperform electricity cost-/carbon-minimizing algorithm SAVING/CARBON (in terms of electricity cost/carbon), we use ALWAYS as the benchmark as it is still widely-used in many performance-driven data centers. Figs. 6(a) and 6(b) show that with the increase in the corresponding weighting factors, both water consumption and carbon emission demonstrate a decreasing trend. This outcome is as expected, since the increase in the weighting factor translates into a higher priority for the corresponding cost during the optimization process. We see in Fig. 6(c) that, with the increase in either water or carbon weight, electricity cost increases because WACE schedules batch jobs to achieve low water consumption and/or carbon emission, while giving less attention to electricity price. Moreover, we observe that WACE has a lower water consumption, carbon emission and electricity cost in comparison to ALWAYS for a wide range of water and carbon weight parameters, demonstrating that WACE can outperform ALWAYS in terms of water consumption, carbon emission and electricity cost at the expense of increasing delay for batch jobs.

4.2.5 *Exploiting temporal diversity*

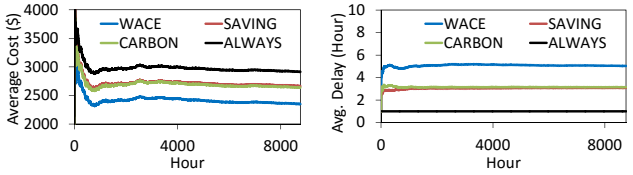
In this section, we show how WACE exploits the temporal variation of electricity price, WUE, and carbon rate to achieve an overall low total cost, whereas the benchmarks do not. As WACE takes the combination of electricity cost, water consumption and carbon footprint into consideration based on a parameterized cost function, we combine the three factors into one "cost factor", defined as "$electricity\_price + water\_weight \times water\_efficiency + carbon\_weight \times carbon\_efficiency$", to better visualize the overall impact of temporal diversity in hourly decisions. Fig. 7 shows a snapshot of 24 hour capacity provisioning and job queue lengths with different scheduling algorithms for the data center in CA. In Fig. 7(a), we see that WACE processes more batch jobs during time slots when the combined cost factor is comparatively low. However, during 12th time slot, although the cost factor is relatively higher, WACE processes more jobs. The reason is that, during time slot 12 the job queue length is too large (shown in Fig. 7(b)),

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing



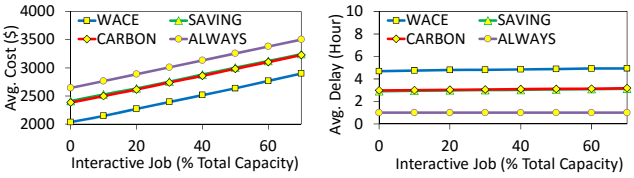
(a) Average combined cost factor (b) Workload distribution

Fig. 8: Workload distribution among data centers.



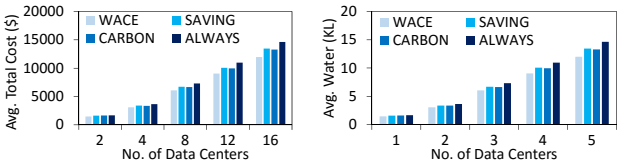
(a) Average total cost (b) Average delay

Fig. 10: Comparison of under Hotmail workload.



(a) Average cost. (b) Delay.

Fig. 9: Impact of interactive job.



(a) (b)

Fig. 11: Impact of number of data centers.

and hence serving those backlogged jobs and reducing the queue length become more essential than optimizing the cost due to delay performance concerns. On the other hand, Fig. 7(c) shows that ALWAYS processes jobs even when the combined cost factor is high (e.g., time slots 6, 11). The job queue for ALWAYS is shown in 7(d), which represents the dispatched batch jobs to the data center in CA. In Fig. 7(e), we see that more jobs are processed during time slots when the electricity price is lower (e.g., time slots 4, 10, 18), since SAVING only cares about electricity price in its optimization. Similar pattern can be observed in Fig. 7(g) for CARBON, where it is evident that more jobs are processed at time slots with low carbon emission rate.

4.2.6 *Exploiting spatial diversity*

Now, we show in Fig. 8 how WACE exploits spatial diversity of electricity price, water efficiency and carbon emission rate, under the same average batch job delay of 5 hours (except for ALWAYS). Fig. 8(b) shows the average amount of batch jobs distributed among the four data centers for four different algorithms. We see that WACE sends relatively more workloads to the data centers in CA and IA than those in IL and NY, because CA and IA have lower combined cost factors, as seen in Fig. 8(a): WACE is concerned about optimizing all the three costs (electricity cost, water consumption and carbon emission), whereas SAVING/CARBON has a different load distribution pattern because SAVING/CARBON only favors electricity cost/carbon efficiency.

4.2.7 *Sensitivity study*

We perform the following two sensitivity studies to evaluate the robustness of WACE.

• **Effect of interactive workload intensity.** Since interactive jobs processed by the data center limits the available server capacity for batch jobs, the amount of interactive jobs (measured based on the peak arrival rate in terms % of total capacity) affects the performance of batch job scheduling algorithms. We choose the same settings as those in Fig. 3, but vary the average intensity of interactive jobs from 0% (i.e., no interactive jobs, only batch jobs) to 70% of total capacity and observe the operational cost and delay performance in Fig. 9. We see in Fig. 9(a) that with increase in interactive job intensity, the average cost increases as expected: increased intensity means that the data centers process more workloads. It can be observed that, WACE still achieves the lowest average total cost, while ALWAYS incurs the highest cost. Fig. 9(b) shows the average delay performance under different interactive workload intensities: batch job delay is not affected much by the change in interactive job intensity. This is because, the peak load of these two job types are often at different time slots and hence even when the data centers process more interactive jobs, they still have enough server capacity to schedule batch jobs with a similar delay.

• **Different workload set.** We conduct our simulation using a different workload trace — Hotmail (shown in Fig. 2), under the same settings as those in Fig. 3. Similar to Fig.3, we compare the average total cost of WACE with SAVING, CARBON and ALWAYS in Fig. 10(a). We see that WACE has a average cost 21% lower than SAVING, 20% lower than CARBON and 30% lower than ALWAYS. Fig. 10(b) shows the delay performance comparison of WACE with the three benchmark algorithms, where it can be seen that WACE has a higher average delay (approximately 1 hour), which is acceptable for batch workloads. The results again demonstrate the capability of WACE in fully taking the advantage of scheduling flexibility of batch jobs to achieve a lower overall cost.

• **Number of data centers.** The four data centers considered in the default case is reasonable, since even leading IT companies like Faceebook and Google have

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing

only a handful of self-managed data centers in the U.S. [44], [45]. Nonetheless, to evaluate WACE in larger systems, we extend our study to more data center locations following similar settings as in the default case. We see in Fig. 11(a) that cost saving increases as we increase the number of data center locations. Similar observation is also made in Fig. 11(b) where water saving increases with the number of data centers. The reason is that as more data centers are included, more spatial-temporal diversities and a higher degree of scheduling freedom can be exploited for more cost saving.

5 Joint Scheduling of Interactive and Batch Jobs

In this section, we extend WACE to incorporate interactive job scheduling decisions. The new online algorithm is called WACE-J (WACE-Joint scheduling of batch and interactive jobs). We present the model for interactive jobs, formulate the problem, and then compare WACE-J with our previous batch job scheduling algorithm WACE (which considers interactive job scheduling as an external decision) through a simulation study.

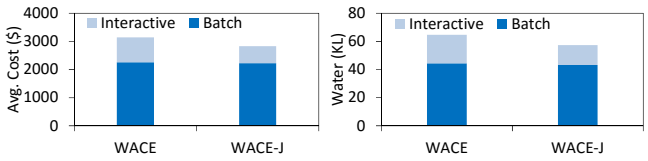
5.1 Model and problem formulation

Model. There are Q regional load balancers, each of which represents a geographically-concentrated source of workloads and then forwards the incoming interactive workloads to the N geo-distributed data centers. We denote the interactive workload arrival rate at the q-th regional load balancer by $\lambda_q(t) = [0, \lambda_{q,max}]$ and the workload is dispatched to data center i at a rate of $\lambda_{i,q}(t)$. We denote by $m_{i,int}(t)$ and $s_{i,int}(t)$ the number of active servers and processing speed for processing interactive workloads at data center i. We include an average delay constraint to ensure performance guarantee of interactive workload and denote the delay threshold by d_{th}. As in the existing literature [1], [46], we use a M/M/1 queue to model the service processing at each server. Specifically, the average service delay for interactive workloads dispatched from load balancer q to data center i is

$$\frac{1}{s_{i,int}(t) - \frac{\sum_{q=1}^{Q} \lambda_{i,q}(t)}{m_{i,int}(t)}} + l_{i,q}(t), \tag{16}$$

where $l_{i,q}(t)$ is the average network latency from load balancer q to data center i which can be approximated based on the distance [1]. As we include interactive job scheduling as part of our decisions, the server power consumption for processing interactive jobs $p_{i,int}(t)$ is no longer an external variable; instead, it can be expressed as

$$p_{i,int}(t) = m_{i,int}(t) \cdot [\alpha_i \cdot s_{i,int}^{n_i}(t) \cdot \frac{\sum_{q=1}^{Q} \lambda_{i,q}(t)}{m_{i,int}(t) s_{i,int}(t)} + p_0], \tag{17}$$



(a) Average total cost. (b) Water consumption.

Fig. 12: Comparison of different algorithms with interactive job scheduling.

where α_i is a positive scaling factor and relates the processing speed to the power consumption, $p_{0,i}$ represents the power consumption in idle or static state, $\frac{\sum_{q=1}^{Q} \lambda_{i,q}(t)}{m_{i,int}(t) s_{i,int}(t)}$ is the utilization of each server in data center i. Then, the total server energy consumption at data center i can be written in the same form as (1), and expressions of all the three types of costs (i.e., electricity cost, water consumption, carbon emission) follow the same as in Section 3.1.

Problem formulation. Following Section 3.2, we present an offline problem formulation for jointly scheduling interactive and batch jobs as follows

$$\textbf{P3}: \quad \min_{\mathcal{D}} \bar{g} = \frac{1}{K} \sum_{t=0}^{K-1} g(\mathbf{a}(t), \lambda(t), \mathbf{m}(t), \mathbf{s}(t)) \tag{18}$$

$$s.t., \quad \text{constraints (8), (9), (10), (11),} \tag{19}$$

$$m_i(t) + m_{i,int}(t) \leq M_i, \ \forall\, i, t \tag{20}$$

$$s_{i,int} \in \mathcal{S}_i = \{s_{i,0}, s_{i,1}, \cdots, s_{i,K_i}\}, \ \forall\, i, t \tag{21}$$

$$\frac{1}{s_{i,int}(t) - \frac{\sum_{q=1}^{Q} \lambda_{i,q}(t)}{m_{i,int}(t)}} + l_{i,q}(t) \leq d_{th}, \ \forall\, i, t \tag{22}$$

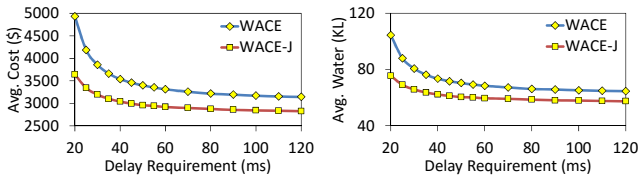
$$\sum_{i=1}^{N} \lambda_{i,q}(t) = \lambda_q(t), \ \forall\, q, t \tag{23}$$

where $\mathcal{D}$ represents a sequence of decisions, i.e., $\mathbf{a}(t), \lambda(t), \mathbf{m}(t), \mathbf{s}(t)$ (also including the resource management decisions for interactive jobs, i.e., $\mathbf{m}_{int}(t), \mathbf{s}_{int}(t)$, which are omitted for brevity), for $t = 0, 1, \cdots, K-1$, which we need to optimize. Compared to the problem formulation in **P1** for batch job scheduling, the additional constraints (20)—(23) in the new formulation **P3** represent: data center capacity, server speed setting, delay performance for interactive jobs, and no workload dropping constraints. To solve **P3**, we can use the same online algorithm as presented in Algorithm 1, with the addition that interactive job scheduling decisions are also optimized at the beginning of each time slot. We omit the details due to space limitations.

5.2 Evaluation

We consider the same settings as used in Section 4 and include one regional load balancer at Denver, CO, which forwards incoming interactive workloads to the four data centers. For brevity, we only compare WACE-J against WACE. Due to the explicit consideration of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing



(a) Average Cost. (b) Water Consumption.

Fig. 13: Impact of delay requirement.

delay performance constraint for interactive jobs, we modify WACE by considering its interactive job scheduling decision as follows: first, incoming interactive jobs are distributed to data centers in proportion to data center capacities; then, servers run at a fixed speed (e.g., medium speed in our study) and then choose the minimum number of servers to satisfy the delay requirement. Fig. 12 shows the comparison in terms of total cost and water consumption between WACE and WACE-J, under the same delay requirement: average delay of five hours for batch jobs and 100ms for interactive jobs. Fig. 12 shows the benefit of jointly scheduling interactive and batch jobs in reducing cost and water consumption, while the benefit in terms of reducing electricity cost and carbon emission is not shown for brevity.

Next, we vary the average delay requirement for interactive jobs and show how it affects WACE-J. As intuitively expected, with a relaxed delay requirement, the total cost of WACE-J will decrease. Figs. 13(a) and 13(b) show the effect of delay requirement on average total cost and water consumption, respectively, for both WACE and WACE-J. In Fig. 13, we observe that WACE-J has a lower average cost and water consumption than those of WACE: WACE-J explicitly optimizes load distribution and capacity provisioning decisions for interactive jobs, while WACE does not utilize this opportunity and hence incurs a higher average cost and water consumption.

6 Related Work

Several prior studies have focused on identifying methods of cost cutting while ensuring the quality-of-service at the same time. For example, finding a balance between energy cost of data center and performance loss through dynamically provisioning server capacity has been the primary focus of many recent studies [42], [47]. Other approaches that are complementary to dynamic capacity provisioning include, but are not limited to, utilizing storage devices to reduce the operational cost of data centers [26], [48], exploiting the spatio-temporal variation of electricity prices [1], [20], [31], [46], [49], and utilizing multiple energy sources (e.g., grid energy, on-site power generation, etc.) [50]. Moreover, there has been considerable interest in reducing carbon footprint through geographical load balancing or "follow the renewables" [1], [4]. Nonetheless, none of the existing literature discusses water consumption reduction in data centers.

Although water consumption has been a critical issue worth addressing, very little research effort has been dedicated to improving water sustainability at data center. Most of the current efforts on water efficiency can be viewed as improved "engineering": for example, installing advanced cooling system [8], using recycled water [19], and reducing indirect water consumption through installation of on-site renewable energy project to scale down electricity consumption [27]. Some other studies that are remotely related to data center water consumption are: developing a dashboard to visualize the water efficiency [51], and pointing out the criticality of water conservation [52]. Some recent studies [22], [23] exploit spatial diversity of water efficiency and geographically schedule interactive workloads among data centers to reduce water footprint, but they neglect the temporal diversity of water efficiency, which allows new water saving opportunities through dynamically scheduling delay-tolerant jobs over time. Another work [53] preliminarily addresses water footprint via online batch job scheduling, but it only considers a single data center and excludes interactive jobs from its decisions.

To sum up, our work extends the previous literature [4], [22], [23], [53], and holistically minimizes electricity cost, carbon emission and water footprint by leveraging the delay tolerance of batch jobs and integrating spatio-temporal diversity of data center water efficiency.

7 Conclusions

In this paper, we addressed the surging water footprint in data centers. We proposed a provably-efficient online batch job scheduling algorithm, WACE, which exploits spatio-temporal diversity of data center water efficiency, carbon rate and electricity price for minimizing the total cost (incorporating electricity cost, water consumption and carbon emission) while bounding the average delay performance. The software-based approach fundamentally differs from the existing water-saving techniques that primarily focus on improved "engineering". We performed a trace-based simulation study to show that WACE reduces the water consumption by over 25% and total cost by 20% compared to state-of-the-art benchmarks, with an acceptable delay increase. Finally, we extended WACE to jointly schedule batch and interactive jobs for further water footprint reduction in geo-distributed data centers.

References

[1] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, "Greening geographical load balancing," in *SIGMETRICS*, 2011.
[2] N. Buchbinder, N. Jain, and I. Menache, "Online job migration for reducing the electricity bill in the cloud," in *IFIP Networking*, 2011.
[3] C. Ren, D. Wang, B. Urgaonkar, and A. Sivasubramaniam, "Carbon-aware energy capacity planning for datacenters," in *MASCOTS*, 2012.

[4] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, "It's not easy being green," *SIGCOMM Comput. Commun. Rev.*, 2012.

[5] T. Evans, "The different technologies for cooling data centers," http://www.apcmedia.com/salestools/VAVR-5UDTU5/VAVR-5UDTU5_R2_EN.pdf.

[6] "AT&T Sustainability," http://www.att.com/gen/landing-pages?pid=24188.

[7] "eBay," http://dse.ebay.com.

[8] "Facebook data center dashboard," http://www.fbpuewue.com.

[9] U.S. Dept. of Energy, "Energy demands on water resources," Dec. 2006.

[10] The Green Grid, "Water usage effectiveness (WUE): A green grid data center sustainability metric," *Whitepaper*, 2011.

[11] Water Facts: Water - Water.org. http://water.org/water-crisis/water-facts/water/.

[12] McKinsey Report, "Charting our water future: Economic frameworks to inform decision-making," 2009.

[13] P. Folger, B. A. Cody, and N. T. Carter, "Drought in the United States: Causes and issues for congress," Apr. 2013, http://www.fas.org/sgp/crs/misc/RL34580.pdf.

[14] Office of Governor in California, "Governor brown declares drought state of emergency," 2014, http://gov.ca.gov/news.php?id=18368.

[15] Uptime Institute, "Data center industry survey," 2013, http://uptimeinstitute.com/2013-survey-results.

[16] U.S. Green Building Council, "Leadership in energy & environmental design," http://www.usgbc.org/leed.

[17] "Federal water efficiency requirements," http://www1.eere.energy.gov/femp/program/waterefficiency_requirements.html.

[18] New York City, "Local Law 84," http://www.nyc.gov/html/gbee/html/plan/ll84_about.shtml.

[19] Google's Data Center Efficiency, "http://www.google.com/about/datacenters/."

[20] M. Lin, Z. Liu, A. Wierman, and L. L. H. Andrew, "Online algorithms for geographical load balancing," in *IGCC*, 2012.

[21] K. Le, R. Bianchini, T. D. Nguyen, O. Bilgir, and M. Martonosi, "Capping the brown energy consumption of internet services at low cost," in *IGCC*, 2010.

[22] S. Ren, "Optimizing water efficiency in distributed data centers," in *Cloud and Green Computing*, 2013.

[23] M. A. Islam, S. Ren, G. Quan, M. Z. Shakir, and A. V. Vasilakos, "Water-constrained geographic load balancing in data centers," *IEEE Trans. Cloud Computing*, 2015.

[24] A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy, "Optimal power allocation in server farms," in *SIGMETRICS*, 2009.

[25] Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang, M. Marwah, and C. Hyser, "Renewable and cooling aware workload management for sustainable data centers," in *SIGMETRICS*, 2012.

[26] R. Urgaonkar, B. Urgaonkar, M. J. Neely, and A. Sivasubramaniam, "Optimal power cost management using stored energy in data centers," in *SIGMETRICS*, 2011.

[27] Apple, "Apple and the environment," http://www.apple.com/environment/.

[28] J. R. Lorch and A. J. Smit, "Improving dynamic voltage scaling algorithms with pace," in *SIGMETRICS*, 2001.

[29] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao, "Energy-aware server provisioning and load dispatching for connection-intensive internet services," in *NSDI*, 2008.

[30] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *ISCA*, 2007.

[31] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, "Cutting the electric bill for internet-scale systems," in *SIGCOMM*, 2009.

[32] L. L. Andrew, M. Lin, and A. Wierman, "Optimality, fairness, and robustness in speed scaling designs," in *SIGMETRICS*, 2010.

[33] D. Wang, C. Ren, A. Sivasubramaniam, B. Urgaonkar, and H. Fathy, "Energy storage in datacenters: what, where, and how much?" in *SIGMETRICS*, 2012.

[34] J. Macknick, R. Newmark, G. Heath, and K. Hallett, "A review of operational water consumption and withdrawal factors for electricity generating technologies," *NREL Tech. Report: NREL/TP-6A20-50900*, 2011.

[35] P. A. Torcellini, N. Long, and R. Judkoff, "Consumptive water use for US power production," 2003.

[36] J. V. Spadaro, L. Langlois, and B. Hamilton, "Greenhouse gas emissions of electricity generation chains: Assessing the difference," *IAEA bulletin*, vol. 42, no. 2, pp. 19–28, 2000.

[37] R. Sharma, A. Shah, C. Bash, T. Christian, and C. Patel, "Water efficiency management in datacenters: Metrics and methodology," in *ISSST*, 2009.

[38] Union of Concerned Scientists, "How it works: Water for power plant cooling," http://www.ucsusa.org/.

[39] M. J. Rutberg, "Modeling water use at thermoelectric power plants," 2012.

[40] "California ISO, http://www.caiso.com/."

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

[42] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," in *IEEE Infocom*, 2011.

[43] PJM RTO, "http://www.pjm.com/."

[44] "Data Center Knowledge: The Facebook Data Center FAQ," http://www.datacenterknowledge.com/the-facebook-data-center-faq/.

[45] "Google: Data center loaction," http://www.google.com/about/datacenters/inside/locations/index.html.

[46] L. Rao, X. Liu, L. Xie, and W. Liu, "Reducing electricity cost: Optimization of distributed internet data centers in a multi-electricity-market environment," in *INFOCOM*, 2010.

[47] B. Guenter, N. Jain, and C. Williams, "Managing cost, performance and reliability tradeoffs for energy-aware server provisioning," in *IEEE Infocom*, 2011.

[48] S. Govindan, D. Wang, A. Sivasubramaniam, and B. Urgaonkar, "Leveraging stored energy for handling power emergencies in aggressively provisioned datacenters," in *ASPLOS*, 2012.

[49] Y. Guo, Z. Ding, Y. Fang, and D. Wu, "Cutting down electricity cost in internet data centers by using energy storage," in *Globecom*, 2011.

[50] W. Deng, F. Liu, H. Jin, C. Wu, and X. Liu, "Multigreen: cost-minimizing multi-source datacenter power supply with online control," in *e-Energy*, 2013.

[51] C. Bash, T. Cader, Y. Chen, D. Gmach, R. Kaufman, D. Milojicic, A. Shah, and P. Sharma, "Cloud sustainability dashboard, dynamically assessing sustainability of data centers and clouds," *HP Labs Tech. Report (HPL-2011-148)*, Sep. 2011.

[52] E. Frachtenberg, "Holistic datacenter design in the open compute project," *Computer*, vol. 45, no. 7, pp. 83–85, Jul. 2012.

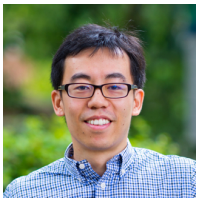
[53] M. A. Islam, K. Ahmed, S. Ren, and G. Quan, "Making data center less "thirsty" via online batch job scheduling," *ICAC*, 2014.

[54] "Online material for WACE," https://goo.gl/PPlaFY.

Mohammad A. Islam is doing his Ph.D. in Electrical Engineering at University of California, Riverside. He received his B.Sc. in Electrical and Electronics Engineering from Bangladesh University of Engineering and Technology in January 2008. He is a member of the Sustainable Computing Group (SCG) at UCR led by Dr. Shaolei Ren. His research interests include data center resource management, cloud computing, and sustainability for IT.

Kishwar Ahmed received his B.Sc. in Computer Science and Engineering from Bangladesh University of Engineering and Technology in October 2009. He is currently working towards the Ph.D. degree at Florida International University, Miami, USA. His research interests include high performance computing, parallel computing, and optimization.

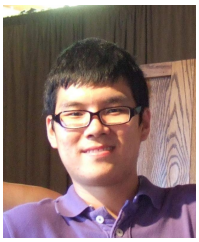
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2535201, IEEE Transactions on Cloud Computing



Hong Xu received the B.Eng. degree from the Department of Information Engineering, The Chinese University of Hong Kong, in 2007, and the M.A.Sc. and Ph.D. 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 was the recipient of 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 IEEE.

Nguyen H. Tran (S10-M11) received the BS degree from Hochiminh City University of Technology and Ph.D degree from Kyung Hee University, in electrical and computer engineering, in 2005 and 2011, respectively. Since 2012, he has been an assistant professor with Department of Computer Science and Engineering, Kyung Hee University. His research interest is to design, analyze and optimize the cutting-edge applications in communication networks, including cloud, mobile-edge computing, datacenters, smart grid, Internet of Things, and heterogeneous, small-cell networks.

Gang Quan (M'02-SM'10) received the B.S. degree from the Tsinghua University, Beijing, China, the M.S. degree from the Chinese Academy of Sciences, Beijing, and the Ph.D. degree from the University of Notre Dame, Notre Dame, IN. He is currently an Associate Professor with the Electrical and Computer Engineering Department, Florida International University, Miami. His research interests includes real-time system, power/thermal aware design, embedded system design, advanced computer architecture and reconfigurable computing. Dr. Quan is the recipient of a National Science Foundation Faculty Career Award. He also won the Best Paper Award from the 38th Design Automation Conference. His paper was also selected as one of the Most Influential Papers of 10 Years Design, Automation, and Test in Europe Conference (DATE) in 2007. Dr. Quan is a senior member of IEEE.



Shaolei Ren received his B.E., M.Phil. and Ph.D. degrees, all in electrical engineering, from Tsinghua University in 2006, Hong Kong University of Science and Technology in 2008, and University of California, Los Angeles, in 2012, respectively. From 2012 to 2015, he was with Florida International University as an Assistant Professor. Since July 2015, he has been an Assistant Professor at University of California, Riverside. His research interests include power-aware computing, data center resource management, and network economics.