EnergyQARE: QoS-Aware Data Center Participation in Smart Grid Regulation Service Reserve Provision

HAO CHEN, YIJIA ZHANG, MICHAEL C. CARAMANIS, and AYSE K. COSKUN,
Boston University

Power market operators have recently introduced smart grid *demand response* (DR), in which electricity consumers regulate their power usage following market requirements. DR helps stabilize the grid and enables integrating a larger amount of intermittent renewable power generation. Data centers provide unique opportunities for DR participation due to their flexibility in both workload servicing and power consumption. While prior studies have focused on data center participation in legacy DR programs such as dynamic energy pricing and peak shaving, this article studies data centers in emerging DR programs, i.e., demand side capacity reserves. Among different types of capacity reserves, *regulation service reserves* (RSRs) are especially attractive due to their relatively higher value. This article proposes **EnergyQARE**, the **Energy** and **Q**uality-of-Service (QoS) **A**ware **R**SR **E**nabler, an approach that enables data center RSR provision in real-life scenarios. EnergyQARE not only provides a bidding strategy in RSR provision, but also contains a runtime policy that adaptively modulates data center power through server power management and server provisioning based on workload QoS feedback. To reflect real-life scenarios, this runtime policy handles a heterogeneous set of jobs and considers transition time delay of servers. Simulated numerical results demonstrate that in a general data center scenario, EnergyQARE provides close to 50% of data center average power consumption as reserves to the market and saves up to 44% in data center electricity cost, while still meeting workload QoS constraints. Case studies in this article show that the percentages of savings are not sensitive to a specific type of non-interactive workload, or the size of the data center, although they depend strongly on data center utilization and parameters of server power states.

CCS Concepts: • **Hardware** → **Smart grid**; **Enterprise level and data centers power issues**; • **General and reference** → *Design*; • **Information systems** → *Data centers*; • **Computing methodologies** → Modeling methodologies;

Additional Key Words and Phrases: Data center, smart grid, power market, demand response, regulation service reserves, power management, workload control, quality of service

ACM Reference format:
Hao Chen, Yijia Zhang, Michael C. Caramanis, and Ayse K. Coskun. 2019. EnergyQARE: QoS-Aware Data Center Participation in Smart Grid Regulation Service Reserve Provision. *ACM Trans. Model. Perform. Eval. Comput. Syst.* 4, 1, Article 2 (January 2019), 31 pages.
https://doi.org/10.1145/3243172

This work is partially supported by Alfred P. Sloan Foundation Grant G-2017-9723, NSF Grant 1733827, and the Boston University College of Engineering Dean's Catalyst Award.

Authors' addresses: H. Chen, Y. Zhang, and A. K. Coskun, Department of Electrical and Computer Engineering, Boston University, 8 Saint Mary's Street, Boston, MA 02215, USA; emails: {haoc, zhangyj, acoskun}@bu.edu; M. C. Caramanis, Division of Systems Engineering, Boston University, 15 Saint Mary's Street, Brookline, MA 02446, USA; email: mcaraman@bu.edu.

2376-3639/2019/01-ART2 $15.00
https://doi.org/10.1145/3243172

1 INTRODUCTION

Fossil fuels, the major source of today's electricity production, do not provide a sustainable energy solution and have tremendous environmental impact. As a result, massive integration of renewables is a worldwide objective. The EU aims to achieve integration of over 20% of renewables in gross energy consumption by 2020 (Bohringer et al. 2009), while in the US, 38 states have set up long-term standards for their renewable portfolios, and 14 states have installed more than 1,000 MW of wind power (AWEA 2015). A growth of the share of renewables to 52% in the US is expected by 2040 (EIA 2014). However, the volatility and intermittency of renewable generation pose significant challenges to independent system operators (ISOs), who need to match supply and demand in power grids in real time. One solution to this challenge is to mitigate the intermittency through energy storage devices (ESDs) (Fooladivanda et al. 2014; Kim et al. 2014; Chen et al. 2015a), which are unfortunately costly. An alternative solution is to harness capacity reserves (Ott 2003; NYISO 2013). While capacity reserves are mainly provided by centralized generators, the market has started to offer incentives for the demand side to provide capacity reserves (PJM 2005; Hansen et al. 2014). Potential demand side reserve providers include smart buildings (Zhang et al. 2014), plug-in hybrid electric vehicles (PHEVs) (Wei et al. 2014; Cui et al. 2015; Li et al. 2014), and data centers (Ghamkhari and Mohsenian-Rad 2012; Goiri et al. 2015).

As data centers are among the fastest growing electricity consumers, their energy efficiency, as well as their potential roles in power markets, is an active area of study. A typical data center today consumes as much energy as 25,000 households (Dayarathna et al. 2016). The total power consumption of data centers in 2014 was estimated at 70 billion kWh, close to 2% of US electrical usage (Shehabi et al. 2016). This power consumption has increased by 4% since 2010 and is expected to increase by another 4% in 2014–2020 (Shehabi et al. 2016). With the increase in power usage as well as the growth of electricity price, the electricity bill of a typical data center tends to double every five years and has become a significant portion of data center expense (Dayarathna et al. 2016). In tandem with the data center growth, advanced server power management techniques such as dynamic voltage frequency scaling (DVFS) (Li and Martinez 2006; Reda et al. 2012) or server idling (Gandhi et al. 2012; Chase et al. 2001), together with flexibility in workload servicing (Ghatikar et al. 2012; Cioara et al. 2018), provide data centers with significant capabilities to modulate their power consumption. Data centers with such built-in capabilities are highly promising candidates to provide demand side capacity reserves. By providing reserves, data centers help enable efficient integration of renewable energy and reduce their electricity bill.

Among all types of capacity reserves, secondary reserves, i.e., regulation service reserves (RSRs) (NYISO 2013), are especially suitable for data centers, not only for the relatively high profits from providing reserves but also because data centers have competitive advantages in offering them. In RSR provision, the demand side is required to dynamically modulate its power consumption to follow an RSR signal broadcast by the ISO every few seconds. Data centers have the inherent capability to regulate their power in the order of seconds (owing to the built-in power states of servers). Though data centers have been widely studied with respect to legacy demand response (DR) programs such as dynamic energy pricing (Liu et al. 2014; Le et al. 2016), peak shaving (Wang et al. 2012; Govindan et al. 2011), and emergency demand reduction (Zhang et al. 2015; Tran et al. 2016; Islam et al. 2016), data center RSR provision is a novel idea. A few studies have also evaluated the capabilities and benefits of RSR provision (Aikema et al. 2012; Chen et al. 2013a; Kirpes and Klingert 2016). However, these studies apply simplified data center models that ignore important characteristics such as heterogeneity in workloads or servers with multiple power states (i.e., transition delay and energy associated with switching server states) (Chen et al. 2015b; Li et al. 2014; Cioara et al. 2018). Others design *offline* policies without considering the hourly-changing power and reserve values in market bidding (Ghasemi-Gol et al. 2014; Aksanli and Rosing 2014). We believe there is a need to design *online* policies of RSR provision with practical data center models that are aware of data center hardware and software characteristics to achieve realistic results. The most relevant work proposes an online "best-tracking" policy, which we have designed in our recent work for the purpose of regulating data center power to track an RSR signal as accurately as possible, however, without considering any workload Quality of Service (QoS) constraints (determined by users or Service Level Agreements (SLAs)) (Chen et al. 2014a). Also, that "best-tracking" policy simply estimates reasonable power and reserve bidding values without optimizing them.

This article proposes EnergyQARE, a data center **Energy** and **Q**oS-**A**ware **R**SR **E**nabler. EnergyQARE is composed of an RSR bidding strategy and a runtime policy that together enable data center RSR provision in real-life scenarios. To the best of our knowledge, EnergyQARE is the first to use a practical data center server and workload model that considers workload heterogeneity and multiple server power states to achieve RSR provision. EnergyQARE not only maximizes the data center monetary savings by bidding substantial levels of reserves (up to 50% of average power consumption) and following RSR signal tracking requirements, but also guarantees tight workload QoS constraints. The main contributions of this article are as follows:

(1) We propose a practical model of the data center in RSR provision, which considers heterogeneities in workload (i.e., different types of applications running), multiple server power states, the associated time delay and energy loss during server state transition, and workload QoS constraints;
(2) We develop a runtime policy as part of EnergyQARE that enables data center to regulate its power to accurately track the RSR signal broadcast every few seconds, while also guaranteeing tight workload QoS constraints. The policy regulates data center power rapidly at fine granularity by dynamic server power management and server provisioning. It can satisfy both signal tracking and QoS constraints as it adaptively makes decisions based on the feedback of the accumulated RSR signal tracking error, workload QoS degradation, and the length of job queues;
(3) We introduce a bidding strategy as part of EnergyQARE that solves a near-optimal energy and reserve bidding problem for data center RSR provision in hour-ahead market to minimize the monetary cost of data center electricity. The proposed hour-ahead market bid is cognizant of the real-time RSR policy employed, and respects both tracking error and QoS constraints with data center clients.

Our results demonstrate that in realistic data center scenarios, EnergyQARE enables the data center to provide up to 50% of its average power consumption as reserves to the power market, rendering massive renewable energy adoption affordable. EnergyQARE also helps the data center save 44% in its electricity bill, while still guaranteeing workload QoS. We conduct case studies on the RSR provision of EnergyQARE in multiple real-life scenarios. The analysis shows that the percentage of monetary savings are not sensitive to job types and the size of data centers, although they depend strongly on data center utilization and parameters of server power states.

The rest of the article is organized as follows: Section 2 describes the power markets, DR programs, and RSR. Section 3 introduces the overall framework of data center RSR provision, as well as the practical server and workload models. Section 4 presents the runtime policy in EnergyQARE that dynamically regulates data center power to track the RSR signal. Section 5 studies the optimal data center energy and reserve bidding strategy. Section 6 presents the simulations and results, followed by discussions in Section 7. Section 8 surveys related work and Section 9 concludes the article.

2 POWER MARKET AND REGULATION SERVICE RESERVES

In this section, we review today's power markets, different types of capacity reserves, and DR programs. Then, we describe the demand side RSR in detail, as we think it is the most suitable and profitable market for data centers participation.

2.1 Capacity Reserves Provisioned in Wholesale Power Markets

Power markets, introduced in the US in 1997 (Ott 2003), have been designed to encourage the competitive participation of centralized generators injecting power to and wholesalers withdrawing power from buses of the high voltage transmission network. Today's power markets clear simultaneously (i.e., they co-optimize) several capacity reserve services and energy. When generation and consumption are not balanced in (near) real-time, power systems may become unstable, leading to catastrophic events such as blackouts. To handle unpredictable disturbances that lead to temporary imbalances, ISOs procure a mix of reserves with different dynamic properties in advance. Each reserve type is characterized by the time scale and the frequency of the associated reserve command deployment. Cascaded short-term markets that are most relevant to this work include day-ahead markets, hour-ahead adjustment markets, and 5-minute close-to-real-time economic dispatch markets (NYISO 2013; Ott 2003). Based on different frequencies of the reserve command deployments, there are primary (i.e., frequency control), secondary (i.e., regulation service) and tertiary (i.e., operating) reserves, for which promises are bid and cleared in the day or hour ahead markets and deployed, respectively, in millisecond, second, and minute intervals (PJM 2017c).

Traditionally, capacity reserves have been offered primarily by centralized generators, but market rules are changing to allow the demand side to offer reserves as well. For example, PJM, one of the largest US ISOs, has allowed electricity loads to participate in reserve transactions since 2006 (PJM 2005), with other ISOs contemplating to follow suit. Demand side capacity reserves, as an emerging type of DR, is starting to play a significant role in stabilizing power systems and is particularly beneficial as intermittent and volatile renewable generation is integrated at ever increasing rates. In the following section, we review both legacy and emerging DR opportunities.

2.2 Demand Response

Demand response (DR) refers to electricity consumers regulating their power usage following market requirements. Widely studied DR programs pertain to a few legacy programs such as dynamic energy pricing (Wierman et al. 2014; Zhu et al. 2013) and peak shaving (Govindan et al. 2011; Wang et al. 2012). In dynamic energy pricing, the demand side modulates its power consumption so as to consume more power at the valley of the energy price and less as prices peak. In addition to charges on energy, medium to large commercial and industrial power consumers are also charged for their peak power over an agreed upon period, e.g., over a month (Govindan et al. 2011). During periods with shortage of supply, there are even stricter limits on the peak power consumption. In these cases, cutting peak power, known as peak shaving (Wang et al. 2012), has been used to reduce costs and enable stability of power systems.

Recently, power markets have started to allow demand side to provide capacity reserves as an emerging DR. In demand side capacity reserves, power consumers complement generators in buying of energy and offering all kinds of capacity reserves in dynamic market. Thus, consumers are obliged to regulate their power consumption to track some dynamic power targets based on the amount of reserve that they have offered (Hansen et al. 2014). As introduced in Section 2.1, there are mainly three types of reserves: frequency control, RSRs, and operating reserves. In *frequency control*, the power targets are determined by local (hence fully distributed) frequency measurements. Reserve providers are required to consume power to counter frequency deviations from

60Hz, and are required to modulate the power consumption near real-time. Today, primary reserves are provided by generators through annual contracts, and there is no short term market for consumers. The targets for *RSR* are determined by an ISO signal, which is used to complement primary control reserves. Unlike the frequency control that is fully distributed, the RSR provider is obligated to follow a centrally broadcast signal updated every few seconds. The clearing prices of RSRs are high and comparable to the price of energy. *Operating reserves* are designed to replenish other reserves, as well as reduce the load burden and the congestion in power systems. The targets of the operating reserves are typically constant low power values maintained for up to a few hours. The speed at which reserves must be offered is far slower than that of primary or secondary reserves (Chen et al. 2014b).

Among these three types of reserves, we believe RSRs are the most suitable and profitable ones for data centers to provide the required regulation speed is in a few seconds, which is achievable using server power controls. The rate of change of power consumption required during the deployment of RSRs is of the order of 0.5% of R per second, where R is the maximal amount of reserves promised at the beginning of the hour. Typical clearing prices of reserves scheduled and hence committed for the duration of an hour vary around the clearing price of energy (PJM 2017a). We focus on typical energy and reserve price conditions by considering reserve clearing prices that are comparable to energy clearing prices. In fact, in anticipation of future trends, one would expect that higher adoption of volatile renewable generation is likely to increase reserve clearing prices. Thus, our choice to use comparable RSR and energy clearing prices is a conservative assumption that likely underestimates monetary savings. The details of demand side RSR provision are introduced in the following section.

2.3 Demand Side Regulation Service Reserve

An RSR provider who has been cleared in the hour ahead market to consume on average $\bar{P}$ power and to provide R reserves with the market clearing prices for energy and reserves at Π^E and Π^R, respectively, pays $\Pi^E \bar{P} - \Pi^R R$, where $\Pi^E \bar{P}$ is the charge for the average power consumption and $\Pi^R R$ is the credit for RSR provision (PJM 2017b, 2017c). To receive the credit, the provider is obliged to modulate its power consumption $P_{con}(t)$ dynamically to track the power target $P_{tgt}(t)$ calculated based on the RSR signal $y(t)$:

$$P_{tgt}(t) = \bar{P} + y(t)R. \tag{1}$$

$y(t)$ is the output of an ISO specified integral proportional filter of the Area Control Error (the difference between actual and scheduled net imports from adjacent balancing areas) and frequency excursions outside a tolerance (i.e., [59.980, 60.020Hz]). This signal is unpredictable and unaffected by any individual market participant. The statistical behavior of $y(t)$, however, is known and can be modeled in advance. $y(t)$ is a zero-mean random variable taking values in the interval $[-1, 1]$, and it follows a well-behaved two-level Markov model whose transition probabilities can be estimated reasonably well (Bilgin et al. 2016). The signal is centrally determined and broadcast every 4s by the ISO, with the increments in each 4s not exceeding $\pm R/(\tau/4)$ where τ is 150s for the fast (F) RSR and 300s for the slower (S) RSR (PJM 2017c). Figure 1 shows actual historical hour-long data trajectories of fast (F) and slow (S) RSR signals that have been recorded in the PJM balancing area.

The signal tracking error is measured during the hour, as

$$\epsilon(t) = \frac{|P_{con}(t) - P_{tgt}(t)|}{R}. \tag{2}$$

PJM and other ISO stakeholders have been discussing the reduction of payments to RSR providers, $\Pi^R R$, by an amount that is proportional to the length of the trajectory that the RSR provider fails to traverse due to poor RSR tracking. For example, if the average error evaluated in Equation (2)

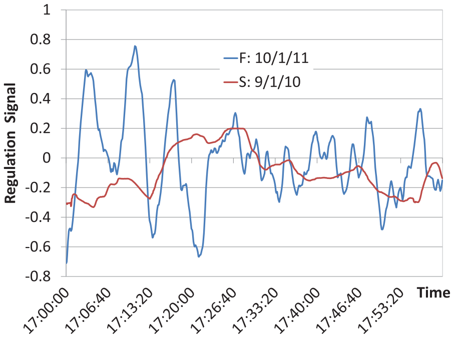


Fig. 1. Typical PJM 150s ramp rate (F) and 300s ramp rate (S) regulation signal trajectories.

is 10%, the RSR provider must return $0.1\Pi^R R$ from its hour ahead revenues (PJM 2017b, 2017c). The tracking error can be therefore monetized and the cost can be modeled in a manner that is consistent with market practice. This quantification is valid as long as the tracking error does not exceed a specified threshold, which, if exceeded, results in the market participant's loss of the right to participate in the regulation reserve market. The reserve provider may lose its contract in further RSR provision, if the tracking error $\epsilon(t)$ exceeds a probabilistic tolerance constraint $(\epsilon^{tol},\ \eta^{\epsilon})$, i.e.,

$$Probability\ \{\epsilon(t) > \epsilon^{tol}\} > 1 - \eta^{\epsilon}, \tag{3}$$

where ϵ^{tol} is the threshold and η^{ϵ} is the probability.

3 THE MODEL OF DATA CENTER IN REGULATION SERVICE RESERVE

A data center[1] system is composed of two major parts: the computational unit and the cooling unit. The computational unit consists of a number of servers,[2] and the cooling unit consists fans, computer room air conditioners, water cooling systems, and so on. Due to the larger thermal time constants involved in thermal dynamics, thus, the cooling unit is not as suitable for RSR as the computational unit. In this work, we study the RSR of the computational unit, while cooling unit can be regulated to participate in slower frequency reserve markets. In the following sections, we first introduce the overall framework of data center RSR provision. Then, we discuss details of the real-life practical models of a single server and the computational unit.

3.1 An Overview of the Data Center RSR Framework

Figure 2 depicts the framework of data center in RSR provision. The central controller of the data center is the unit responsible for (a) communicating with the power market, (b) optimizing and making decisions in server and workload controls, and (c) monitoring the performance. Specifically, the central controller repeats the following steps for each hour (we assume an hour-ahead RSR market participation):

(1) The central controller solves the optimal RSR capacity planning problem; i.e., determine the average power consumption $\bar{P}$ and the reserve provision R based on statistical information of workload arrivals for the next hour. The workload is forecast based on historical patterns if the real-time information is not available ahead.

[1] We define data centers broadly including both enterprise and high performance computing data centers.

[2] The computational unit also includes networking, storage, uninterruptible power supply (UPS) and other elements. This article focuses on providing RSR using server-level controls.

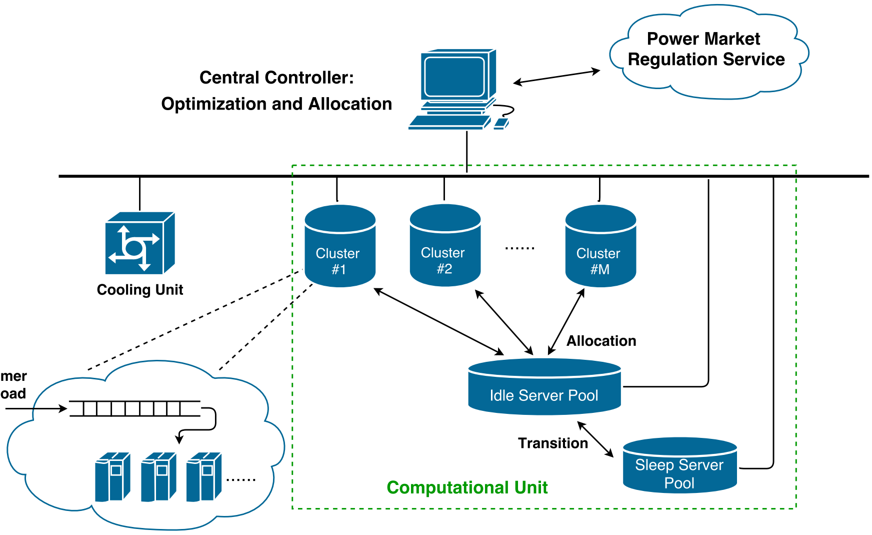


Fig. 2. The framework of a data center in power market RSR participation.

(2) The central controller then bids $(\bar{P}, R)$ in the power market. Once the bid is approved, ISO dynamically broadcasts RSR signal $y(t)$ in real time.
(3) The central controller is then required to modulate the power consumption of the data center to track the power $\bar{P} + y(t)R$ calculated by the signal $y(t)$. At runtime, we dynamically classify servers into multiple clusters based on the types of jobs that they are serving, with each cluster containing servers serving the same type of jobs. With this model, though the data center receives heterogeneous workload, each cluster serves a homogeneous set of jobs. This model is, in principle, similar to the design of today's high performance computing (HPC) clusters, where dedicated and optimized sets of servers are assigned to specific jobs. The number of servers in each cluster and their power budget are dynamically adjusted by the central controller to track the RSR signal and maintain QoS of the workloads.
(4) In each cluster, jobs arrive in a queue and wait there until scheduled. Servers run jobs at power states assigned by the central controller. The controller also continuously monitors the RSR signal tracking error, the length of job queue in each cluster, and the QoS of each job type and uses this information as feedback to adaptively update server allocation and power budgeting decisions of each cluster.

The above steps indicate two key problems to be solved: (1) finding an efficient runtime policy that regulates the data center power to accurately track the RSR signal, while also guaranteeing workload QoS; (2) designing a power and reserve bidding strategy that minimizes the data center energy costs. The proposed EnergyQARE solution in this article aims at solving these two problems on a practical data center RSR provision model. Our runtime policy in EnergyQARE is introduced in Section 4 and our proposed bidding strategy is discussed in Section 5.

3.2 The Server Model

Servers in data centers can be put into different power states. Typical states include active (i.e., running), idle, sleep, and shut down. We use $P_s(t)$ to denote the power consumption of an active server s at time t. This power consumption can be regulated dynamically. There are a number of

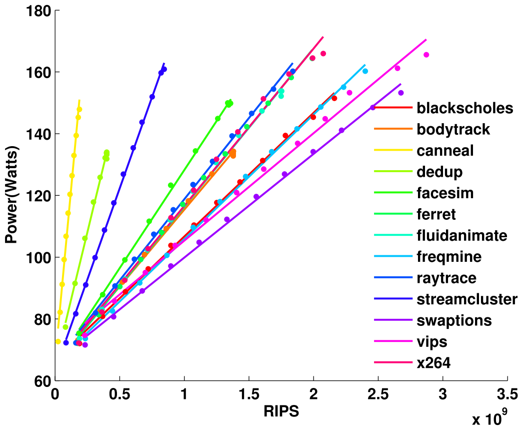


Fig. 3. The relation between power and service rate (in RIPS) for applications in PARSEC-2.1 benchmark suite. Dots are the real measurements and lines are the linear fit curves.

dynamic power management techniques to regulate the power consumption of active servers, such as DVFS (Li and Martinez 2006; Reda et al. 2012) and setting CPU resource limits (Chen et al. 2013b).

The server throughput, i.e., the service rate $u_s(t)$ of the server s at time t, is affected by the power $P_s(t)$. Prior studies have demonstrated a linear relation between power and the service rate for active servers (Dayarathna et al. 2016; Chen et al. 2013b; Tuncer et al. 2014), i.e.,

$$P_s(t) = \alpha_j u_s(t) + P_{idle,s}, \tag{4}$$

where α_j is the coefficient depending on the job type j and $P_{idle,s}$ is the idle power of the server s. To examine the linearity of the relation, we conduct experiments on applications from PARSEC-2.1 suite (Christian 2011) on a 1U server that has an AMD Magny Cours (Opteron 6172) processor, with 12 cores on a single chip. The server is virtualized by VMware vSphere 5.1 ESXi hypervisor, and we use the CPU resource limits knob to regulate the power (Chen et al. 2013b). Tuning the CPU resource limits adjusts the server power and the throughput by changing the resources allocated to the virtual machine (VM). It is a desirable control knob for RSR provision as it can quickly modulate the power at a fine granularity (Chen et al. 2013b). Figure 3 shows power vs. throughput curve on our server. A linear fit between server power and the service rate (represented by retired instructions per second, i.e., RIPS) provides less than 5% mean square error for all the applications in PARSEC-2.1 suite.

A server is "idle" if it is turned on but not servicing any job. Idle servers can be put into the "sleep" state to save power. The power consumption of a server in idle and sleep states are both constant, denoted as $P_{idle,s}$ and $P_{slp,s}$, respectively, for server s, with $P_{idle,s} > P_{slp,s}$. In this work, we assume homogeneous servers, and thus we omit the notation s for simplicity. There are time delays and energy loss when a server is suspended to or resumed from the sleep state. The time delay and energy loss of suspending a server to sleep is usually small, while those of resuming a server from the sleep state is notable and requires explicit consideration (Gandhi et al. 2012; Isci et al. 2013). We denote this server transition time as T_{tran}. During the transition period, the power consumption of the server is denoted as P_{tran}, which is often constant and close to the maximal server power (Isci et al. 2013).

A server can also be completely "shut down" and consume zero power. However, servers are rarely shut down in today's data centers due to the very large time delay and energy loss of resuming from a fully shut down state. Thus, we do not consider the shut down state in this work.

3.3 The Model of the Computational Unit

As shown in Figure 2, in our data center model, servers in the computational unit are classified into several sub-units: (1) an idle server pool, (2) a sleeping server pool, (3) several active server clusters. Active servers are classified into clusters according to the job type they are serving, with each cluster of servers running a homogeneous set of jobs. Servers in the idle server pool are assigned by the central controller to active clusters as needed. Active servers are immediately released back to the idle server pool if they finish their jobs. The number of servers in each cluster, as well as their power budget, are dynamically modulated by the central controller based on multiple system states such as the job queue length in each cluster, the RSR signal and the QoS constraint of each job type, and so on. The sleeping server pool only interacts with the idle server pool. We ignore the time delay in transition from idle to sleep, since it is relatively short compared to the server wake up delay. However, if sleeping servers are resumed, they are turned into the transition state for a certain time before they become idle and join in the idle server pool.

In each cluster, we assume there is a first-come-first-served (FCFS) queue for holding the incoming jobs submitted by the users. Once a job arrives, it is put into the queue and waits to be scheduled for service. We use the FCFS queue, because it is simple but efficient and is one of the most widely used scheduling policies in today's systems. Moreover, as each cluster contains only homogeneous workload, other scheduling polices such as shortest job first and shortest remaining processing time (Yang et al. 2012) do not provide additional benefits. In addition, we assume that each server can only serve one job a time; thus, we do not consider server consolidation. In fact, servers in HPC data centers typically run one job at a time for performance and privacy reasons. We also assume that a running job is not allowed to be stalled or preempted.

Workloads in data centers mainly fall into two catalogs: (1) interactive jobs such as email clients, web search, stock transactions, and so on, which are highly sensitive to the latency, and (2) batch jobs that are more tolerable to latency. Some of the batch jobs can be accumulated and run later when there is availability in the data center (Liu et al. 2012; Verma et al. 2015). In this work, we focus on data center clusters with batch jobs in RSR provision. Since batch jobs provide more flexibilities in QoS constraints, clusters with them are more suitable for RSR provision.

4 ENERGYQARE RUNTIME POLICY

To participate in the RSR market, data centers need to dynamically modulate their power consumption to track the RSR signal, while at the same time guaranteeing QoS in workload servicing to meet the SLAs. However, there is often a tradeoff between RSR signal tracking and QoS. For example, a low RSR signal value limits the data center power budget and degrades the workload QoS. In this section, we introduce the EnergyQARE runtime policy that efficiently searches for a balance between performance and tracking error. EnergyQARE dynamically monitors signal tracking and workload servicing and adaptively makes decisions on which of them (i.e., tracking error or QoS) should be given higher priority.

4.1 An Overview of the EnergyQARE Runtime Policy

The flowchart of the EnergyQARE runtime policy is shown in Figure 4. The principle is to dynamically make decisions based on monitored signal tracking and workload QoS state variables. At each time t, the monitored main state variables include the following:

(1) the average of tracking error, i.e., $\bar{\epsilon}(t)$ during the time window $[t-T, t]$;
(2) the average of QoS degradation, i.e., $\bar{D}_j(t)$ for each cluster j in $[t-T, t]$;

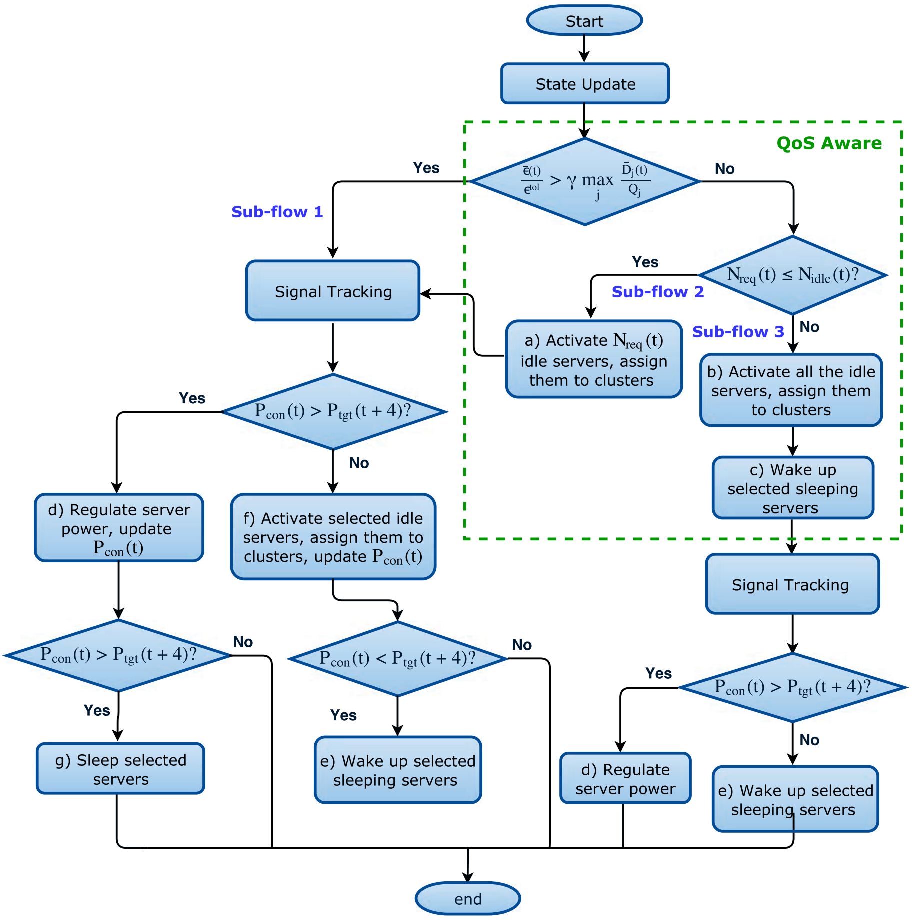

Start
State Update
QoS Aware
$\frac{\bar{e}(t)}{e^{tol}} > \gamma \max_j \frac{\bar{D}_j(t)}{Q_j}$
Yes
No
Sub-flow 1
Signal Tracking
$N_{req}(t) \le N_{idle}(t)$?
Yes
Sub-flow 2
Sub-flow 3
No
a) Activate $N_{req}(t)$ idle servers, assign them to clusters
b) Activate all the idle servers, assign them to clusters
c) Wake up selected sleeping servers
$P_{con}(t) > P_{tgt}(t+4)$?
Yes
No
d) Regulate server power, update $P_{con}(t)$
f) Activate selected idle servers, assign them to clusters, update $P_{con}(t)$
Signal Tracking
$P_{con}(t) > P_{tgt}(t+4)$?
No
Yes
$P_{con}(t) < P_{tgt}(t+4)$?
No
Yes
$P_{con}(t) > P_{tgt}(t+4)$?
Yes
No
g) Sleep selected servers
e) Wake up selected sleeping servers
d) Regulate server power
e) Wake up selected sleeping servers
end

Fig. 4. The flowchart of the EnergyQARE runtime policy. Actions are denoted by letters a, b, c, . . . g. Similar actions are denoted by the same letter.

(3) the total number of additional required active servers $N_{req}(t)$ to meet the SLAs based on the Little's Law (Leon-Garcial 2008), which is highly correlated with the length of the job queue;
(4) the states of jobs and servers;
(5) the value of RSR signal $y(t)$.

Based on these dynamically monitored state variables, EnergyQARE selects one or more from the following actions:

(1) wake up sleeping servers;
(2) put idle servers into sleep;
(3) assign idle servers to clusters to serve waiting jobs;
(4) regulate the power and service rate of the active servers.

In the following sections, we introduce the details of the state variables and actions, as well as the overall runtime policy. All symbols are listed and described in Table 1.

Table 1. Description of Symbols

$\epsilon(t)$	Instant tracking error at time t.
$\bar{\epsilon}(t)$	Average of the instant tracking error in $[t-T, t]$.
$(\bar{\epsilon}, \sigma_\epsilon)$	Mean and standard deviation of the instant tracking error during the hour.
$(\epsilon^{tol},\ \eta^\epsilon)$	Probabilistic constraint on RSR signal tracking.
Π^E	Market clearing price on energy.
Π^R	Market clearing price on reserves.
Π^ϵ	Price on RSR signal tracking error.
$D_j(i)$	QoS degradation of job i in cluster j.
$\bar{D}_j(t)$	Average QoS degradation of all jobs finished in $[t-T, t]$ in cluster j.
$(\bar{D}_j, \sigma_{D_j})$	Mean and standard deviation of the QoS degradation of jobs in cluster j during the hour.
I_i	The total number of instructions to be executed in job i.
$I_{r,i}(t)$	The number of residual instructions of job i at time t.
M	The number of clusters (job types) in the data center.
N	The total number of servers in the data center.
$N_j(t)$	The number of servers in cluster j at t.
$N_{req,j}(t)$	The number of additional servers required for cluster j at t.
$N_{req}(t)$	The total number of additional active servers required for all clusters at t.
$N_{idle}(t)$	The number of servers in the idle pool at t.
$N_{rdslp}(t)$	The number of servers that are ready to be put into sleep at t.
$N_{slp}(t)$	The number of servers in the sleep pool at t.
$N_{tran}(t)$	The number of servers in the transition state at t.
$N_{up}(t)$	The number of servers to be woken up at t.
$N_{toslp}(t)$	The number of servers to be put into sleep at t.
$N_{assg,j}(t)$	The number of idle servers that are assigned to cluster j at t.
$N_{assg}(t)$	The total number of idle servers that are activated and assigned to all clusters at t.
$\bar{P}$	Hour ahead bidding of the average power consumption.
P_{Smax}	The maximal possible server power.
$P_j(t)$	The power of active servers in cluster j at time t.
P_{idle}	Server power in the idle state.
P_{tran}	Server power during the transition state.
P_{slp}	Server power in the sleep state.
$P_{con}(t)$	Real data center power consumption at t.
$P_{tgt}(t)$	Targeted power at t calculated based on the RSR signal and the biddings.
$(Q_j,\ \eta_j)$	SLA probabilistic contraint for jobs in cluster j.
R	Hour ahead bidding of the reserve provision.
T	Size of the feedback time window.
$T_{min}(i)$	The shortest possible processing time of job i.
$T_{min,j}$	The shortest possible processing time of jobs in cluster j.
$T_{sys}(i)$	Job system time (waiting time plus processing time) of job i.
$\bar{T}^L_{sys,j}(t)$	Average job system time for cluster j at t based on Little's Law.
T_{tran}	Time delay for waking up a server.
$T_{w,s}(t)$	Time that a server s has been in the transition state at t.
$T_{idle,s}(t)$	Time that a server s has been in the idle state at t.
T_{out}	The timeout value used to determine whether to sleep a server.

(Continued)

Table 1. Continued

$p_j(t)$	The number of running jobs in cluster j at t.
$q_j(t)$	The number of waiting jobs in cluster j at t.
$u_j(t)$	Service rate of servers in cluster j at t.
$u_{max,j}$	The maximal possible service rate for workload in cluster j.
$u_{min,j}(t)$	The minimal service rate for job in cluster j at t to meet SLAs.
$y(t)$	The RSR signal at t.

4.2 State Variables in EnergyQARE

The system states are measured and updated at the beginning of each time interval t. The main state variables used for decision making in EnergyQARE are as follows:

A. The Average Tracking Error $\bar{\epsilon}(t)$. $\bar{\epsilon}(t)$ is an average of the instant signal tracking error $\epsilon(t)$ (calculated in Equation (2)) during a past time window $[t-T, t]$, i.e.,

$$\bar{\epsilon}(t) = \frac{1}{T}\sum_{\tau=t-T}^{t} \epsilon(\tau). \tag{5}$$

The size of the time window T is adjustable.

B. The Average QoS Degradation $\bar{D}_j(t)$. We first define QoS degradation $D(i)$ for a single job i as

$$D(i) = \frac{T_{sys}(i) - T_{min}(i)}{T_{min}(i)}, \tag{6}$$

where $T_{sys}(i)$ is the job system time (i.e., job waiting time plus job processing time), and $T_{min}(i)$ is the shortest possible processing time of job i, which refers to the time of servicing the job without power restrictions and without waiting time. In Equation (6), $D(i)$ is a normalized value (to the shortest processing time) and hence is unified among different types of jobs. Smaller $D(i)$ represents better workload QoS, and $D(i) = 0$ means no degradation in the service of job i.

Similar to the way that we calculate $\bar{\epsilon}(t)$, we calculate the average QoS degradation $\bar{D}_j(t)$ for each cluster j at time t as an average of $D(i)$ for all jobs i in cluster j that have finished their servicing in a past time window $[t-T, t]$:

$$\bar{D}_j(t) = \frac{\sum_{i=1}^{N_{F,j}(t)} D(i)}{N_{F,j}(t)}, \tag{7}$$

where $N_{F,j}(t)$ is the number of jobs that finished in time window $[t-T, t]$ in cluster j.

C. The Total Number of Additional Active Servers Required for All Clusters $N_{req}(t)$. We first calculate the number of additional servers required by each cluster j, i.e., $N_{req,j}(t)$, based on Little's Law (Leon-Garcial 2008). We denote the number of waiting jobs and running jobs in cluster j as $q_j(t)$ and $p_j(t)$ at time t, respectively. Since each cluster contains homogeneous workload, we assume all servers in each cluster run at the same service rate $u_j(t)$, to ensure fairness in servicing the same type of jobs. Then, the average job system time calculated from Little's Law for cluster j at time t is

$$\bar{T}^{L}_{sys,j}(t) = \frac{q_j(t) + f_r(p_j(t))}{u_j(t)n_j(t)}, \tag{8}$$

where $n_j(t)$ is the number of servers in cluster j at time t. Superscript L represents that this average system time is calculated from Little's Law, which distinguishes it from the notation $\bar{T}_{sys,j}(t)$, i.e., the average system time from real measurement. Function $f_r(\cdot)$ transforms the unfinished parts of running jobs to the number of full jobs, i.e.,

$$f_r(p_j(t)) = \sum_{i=1}^{p_j(t)} \frac{I_{r,i}(t)}{I_i}, \tag{9}$$

where $I_{r,i}(t)$ is the number of unfinished instructions of job i at time t, I_i is the total number of instructions of the job i. A job can be considered as a set of instructions, and finishing a job is equivalent to execute all instructions of that job.

The average job system time $\bar{T}^L_{sys,j}(t)$ is constrained by SLAs. SLAs today are mainly defined as the availability of the service; e.g., 99.9% of the time the service is guaranteed to be available (Amazon 2013). Customers and service operators start to include performance measures of the service, e.g., the throughput, or the delay of service, into SLAs (Ghamkhari and Mohsenian-Rad 2012). In this article, we design the SLAs based on the job system time, as it is a key measure to evaluate the job servicing performance. For jobs in cluster j, we define SLAs in a probabilistic form:

$$Probability \left\{ \frac{T_{sys}(i) - T_{min,j}}{T_{min,j}} \leq Q_j \right\} \geq \eta_j, \tag{10}$$

where $T_{min,j}$ is the shortest possible job processing time for jobs in cluster j. Here we use $T_{min,j}$ to substitute the $T_{min}(i)$ in Equation (6), because we have assumed that each cluster contains homogeneous jobs that have the same $T_{min}(i)$. $(Q_j, \ \eta_j)$ are thresholds in the SLA constraint. Equation (10) represents that jobs in cluster j are required to be served with the QoS smaller than Q_j at a probability larger than η_j. From Equation (10), we deduce a constraint on the average system time $\bar{T}^L_{sys,j}(t)$ as

$$\bar{T}^L_{sys,j}(t) \leq (\beta Q_j + 1) \cdot T_{min,j}, \tag{11}$$

where β is the coefficient determined by the probability distribution of $T_{sys}(i)$. Putting Equation (8) into Equation (11), we obtain a constraint on the number of servers in cluster j:

$$n_j(t) \geq \frac{q_j(t) + f_r(p_j(t))}{(\beta Q_j + 1) \cdot T_{min,j} u_j(t)}. \tag{12}$$

Prior work has shown that servicing jobs at the maximal possible service rate u_{max} in general provides best energy efficiency (Gandhi et al. 2012; Chen et al. 2014b). Hence in this work we assume that by default all the jobs are served at their maximal possible rate, and we use $u_{max,j}$ to substitute $u_j(t)$ in Equation (12). Since $u_{max,j} \cdot T_{min,j} = 1$ (recalling that $T_{min,j}$ is the shortest possible processing time when the job is serviced at the highest service rate $u_{max,j}$), Equation (12) is simplified to

$$n_j(t) \geq \frac{q_j(t) + f_r(p_j(t))}{\beta Q_j + 1}. \tag{13}$$

The right-hand side of Equation (13) is the minimal number of required servers for cluster j at time t, to meet the system time constraint from Little's Law. Assuming currently there are $N_j(t)$ active servers in cluster j, the number of additional servers required for cluster j to meet the QoS constraint is

$$N_{req,j}(t) = max \left\{ 0, \ \ \frac{q_j(t) + f_r(p_j(t))}{\beta Q_j + 1} - N_j(t) \right\}. \tag{14}$$

The total number of required servers for all the clusters at time t, i.e., $N_{req}(t)$, is the summation of the number of the required servers in all clusters:

$$N_{req}(t) = \sum_{j=1}^{M} N_{req,j}(t). \tag{15}$$

In addition to these listed three state variables, the state of each job and server, as well as the RSR signal value $y(t)$ are directly monitored and recorded.

4.3 The Overall EnergyQARE Runtime Policy

In this section, we explain the EnergyQARE runtime policy shown in the flowchart in Figure 4. In the figure, we mark each action with a letter. Similar actions are marked with the same letter.

At the beginning of each time interval t, the system states are updated by

- Checking whether any servers have finished their jobs. Servers that finished jobs are put back into the idle server pool. The QoS of finished jobs is calculated;
- Checking whether any servers have finished their transition processes. Servers that have finished their transition processes are put into the idle server pool;
- Monitoring new coming jobs in the waiting queue for each cluster.

After these updates, we calculate major state variables introduced in Section 4.2. Then we compare the average tracking error (normalized by its tolerance ϵ^{tol} in Equation (3)), i.e., $\frac{\bar{\epsilon}(t)}{\epsilon^{tol}}$, with the average QoS degradation (normalized by its tolerance Q_j in Equation (10)), i.e., $\gamma \max_j \frac{D_j(t)}{Q_j}$. Here $\max_j \frac{\bar{D}_j(t)}{Q_j}$ represents the worst average QoS degradation (normalized by the tolerance Q_j) among all clusters at time t. γ is the coefficient given to the QoS degradation in comparison with the tracking error. Different γ indicates different focuses in terms of tracking error and workload QoS, e.g., a small γ can be used if tracking error is more cared about. In our simulations, we always set $\gamma = 1$. We also compare the number of required servers $N_{req}(t)$ with the number of idle servers $N_{idle}(t)$. We move from the main flow to one of the three sub-flows based on these two comparisons. The three sub-flows are

(1) $\frac{\bar{\epsilon}(t)}{\epsilon^{tol}} > \gamma \max_j \frac{\bar{D}_j(t)}{Q_j}$. In this scenario, the signal tracking performance is relatively worse than the QoS performance, so the policy selects actions to track the signal accurately. The real power $P_{con}(t)$ and the targeted power $P_{tgt}(t+4)$ are calculated and compared. If $P_{con}(t) > P_{tgt}(t+4)$, i.e., the real power needs to be reduced, then we first slow down some servers in Action d. If we still have $P_{con}(t) > P_{tgt}(t+4)$, then we put some servers into sleep in Action g. We put Action d before Action g to reduce server transitions that bring time delay and energy loss. However, if $P_{con}(t) < P_{tgt}(t+4)$, i.e., the real power needs to be increased, then we first activate idle servers and assign them to clusters in Action f. After that, if we still have the condition $P_{con}(t) < P_{tgt}(t+4)$, we wake up some sleeping servers in Action e. Similarly, we give the server transition action (in Action e) a low priority due to the concern of additional energy loss and time delay.

(2) $\frac{\bar{\epsilon}(t)}{\epsilon^{tol}} \leq \gamma \max_j \frac{\bar{D}_j(t)}{Q_j}$, and $N_{req}(t) \leq N_{idle}(t)$. In this scenario, the QoS performance is relatively worse than the signal tracking performance, but the current number of idle servers is sufficient to improve QoS to meet the requirement. We first activate and assign $N_{req}(t)$ idle servers to all clusters to serve waiting jobs in Action a. Then we focus on the signal tracking and move back to Sub-flow 1.

(3) $\frac{\bar{\epsilon}(t)}{\epsilon^{tol}} \leq \gamma \max_j \frac{\bar{D}_j(t)}{Q_j}$ and $N_{req}(t) > N_{idle}(t)$. In this scenario, the QoS performance is relatively worse than the signal tracking performance, and the number of idle servers is not sufficient to meet the requirement. We first activate all idle servers with waiting jobs in Action *b*. Then, we wake up additional servers so as to meet the requirement $N_{req}(t)$ in Action *c*. After that, we consider signal tracking. If $P_{con}(t) > P_{tgt}(t+4)$, then we slow down servers in Action *d*. At this step, we no longer consider to put servers to sleep, because prior to this action we have already been required to wake up servers in Action *c*. However, if $P_{con}(t) < P_{tgt}(t+4)$, since all the idle servers have been activated in Action *b*, the only action to increase $P_{con}(t)$ is to wake up additional sleeping servers in Action *e*.

The details of each action will be introduced in the next section.

4.4 Detailed Actions in EnergyQARE

All the actions in Figure 4 fall into four basic groups introduced in Section 4.1: Wake up sleeping servers (Actions *c* and *e*), put idle servers into sleep (Action *g*), assign idle servers to clusters to serve waiting jobs (Actions *a*, *b* and *f*), and regulate the power and service rate of active servers (Action *d*). We discuss each of them in detail below.

A. Wake Up Sleeping Servers. In Figure 4, both Actions *c* and *e* wake up some sleeping servers. Action *c* is called due to a “QoS crisis” (i.e., QoS of jobs in some clusters tends to violate the SLAs), and the number of idle servers at time *t* is smaller than the number of required servers. Therefore, additional servers are required to be woken up to increase QoS. The number of servers to be woken up, i.e., $N_{up}(t)$ is determined as

$$N_{up}(t) = min\{N_{req}(t) - N_{idle}(t) - f_{eqv}(N_{tran}(t)), N_{slp}(t)\}, \tag{16}$$

where $N_{slp}(t)$ is the number of servers in sleep state at *t*. In Equation (16), we also take the number of servers in the transition state, i.e., $N_{tran}(t)$, into account by a function $f_{eqv}(\cdot)$. This function calculates the equivalent number of idle servers that the number of servers in transition is, based on the time that these servers have spent in the transition state. Specifically, assuming that a server *s* has been in the transition state for $T_{w,s}(t)$ seconds at time *t*, and the waking up process takes T_{tran} seconds, then this server *s* is equivalent to $T_{w,s}(t)/T_{tran}$ idle server. Hence,

$$f_{eqv}(N_{tran}(t)) = \frac{\sum_{s=1}^{N_{tran}(t)} T_{w,s}(t)}{T_{tran}}. \tag{17}$$

In Action *e*, $N_{up}(t)$ is determined for the purpose of better signal tracking. Waking up a server can increase its power from P_{slp} to P_{tran}. Hence,

$$N_{up}(t) = min\left\{\frac{P_{tgt}(t+4) - P_{con}(t)}{P_{tran} - P_{slp}}, N_{slp}(t)\right\}, \tag{18}$$

where P_{tran} and P_{slp} are the power consumption of the server in the transition state and in the sleeping state, respectively.

B. Put Idle Servers into Sleep. In Action *g*, the number of idle servers to be put into sleep state, i.e., $N_{toslp}(t)$, is determined for the purpose of achieving better signal tracking, as putting an idle server into the sleep state can reduce its power from P_{idle} to P_{slp}. Therefore,

$$N_{toslp}(t) = min\left\{\frac{P_{con}(t) - P_{tgt}(t+4)}{P_{idle} - P_{slp}}, N_{rdslp}(t)\right\}, \tag{19}$$

where P_{idle} is the server idle power, $N_{rdslp}(t)$ is the number of servers that are ready to be put into the sleep state at time t. $N_{rdslp}(t)$ is introduced to avoid waking up servers or putting servers into sleep over-frequently. We apply a "timeout" mechanism, in which only when a server has been idle for a time longer than T_{out}, it can be put into the sleep state. Prior work introduces a way to determine the T_{out} from the energy efficiency perspective (Gandhi et al. 2012), and they suggest

$$T_{out} = \frac{P_{tran} \cdot T_{tran}}{P_{idle}}. \tag{20}$$

We use $T_{idle,s}(t)$ to denote the time that the server s has been in the idle state till t. If $T_{idle,s}(t) > T_{out}$, then the server s is ready to be put into the sleep state. Therefore,

$$N_{rdslp}(t) = \sum_{s=1}^{N_{idle}(t)} \mathbb{I}_{\{T_{idle,s}(t) > T_{out}\}}. \tag{21}$$

C. Assign Idle Servers to Clusters to Serve Waiting Jobs. If there are both waiting jobs and idle servers in the system, then we can assign idle servers to clusters to serve some of the waiting jobs.[3] Since all M clusters share the idle server pool, the number of idle servers assigned to each cluster j, i.e., $N_{assg,j}(t)$ needs to be determined. The first step is to determine the total number of idle servers to be assigned, i.e., $N_{assg}(t)$. In Figure 4, Actions a and b are called because of the "QoS crisis," and hence $N_{assg}(t)$ is calculated based the QoS requirement. In Action a, since the number of idle servers is not smaller than the number of required servers, i.e., $N_{idle}(t) \geq N_{req}(t)$, we simply activate and assign $N_{req}(t)$ servers to clusters, i.e., $N_{assg}(t) = N_{req}(t)$. In Action b, since $N_{idle}(t) < N_{req}(t)$, i.e., the idle servers are not sufficient, thus all the idle servers at time t require to be activated, i.e., $N_{assg}(t) = N_{idle}(t)$.

Differing from Actions a and b, in Action f, $N_{assg}(t)$ is determined for the purpose of achieving better signal tracking, hence $N_{assg}(t)$ is calculated as

$$N_{assg}(t) = min\left\{\frac{P_{tgt}(t+4) - P_{con}(t)}{P_{Smax} - P_{idle}},\ N_{idle}(t),\ q(t)\right\},$$

where P_{Smax} is the server maximal power, $q(t)$ is the total number of jobs in all the queues at time t, i.e., $q(t) = \sum_{j=1}^{M} q_j(t)$. The equation represents that the number of activated servers must be smaller than the current number of idle servers $N_{idle}(t)$ and the total number of jobs in the queue $q(t)$. In Actions a and b we do not need to consider $q(t)$ again, because $N_{req}(t)$ is calculated based on $q(t)$ in Equation (14). Overall we have

$$N_{assg}(t) = \begin{cases} N_{req}(t), & \text{in Action } a, \\ N_{idle}(t), & \text{in Action } b, \\ min\left\{\frac{P_{tgt}(t+4)-P_{con}(t)}{P_{Smax}-P_{idle}},\ N_{idle}(t),\ q(t)\right\}, & \text{in Action } f. \end{cases} \tag{22}$$

Next, we determine the number of idle servers assigned to each cluster j. First, the number of assigned servers to each cluster j, i.e., $N_{assg,j}(t)$ should not be larger than the number of jobs waiting in the queue in that cluster, i.e., $q_j(t)$,

$$\begin{array}{l} \sum_{j=1}^{M} N_{assg,j}(t) = N_{assg}(t), \\ N_{assg,j}(t) \leq q_j(t),\ \forall j. \end{array} \tag{23}$$

[3] If the signal tracking has a higher priority than the workload QoS and the power cap determined by our policy is not sufficient to support additional active servers, even if there are both waiting jobs and idle servers in the system, then idle servers are not activated for job servicing.

There are many different ways to allocate idle servers to each cluster. Simple methods include round robin, random, or shortest job first. From experiments, we find that an "urgent QoS first" strategy that allocates idle servers based on their QoS constraints can provide better performance in both QoS and signal tracking than other methods in our scenario. Specifically, the "urgent QoS first" strategy is as follows: the average QoS degradation of each cluster j in the past time window $[t-T, t]$, i.e., $\bar{D}_j(t)$ is first calculated by Equation (7); then, we calculate the weight for each cluster:

$$w_j(t) = \frac{\bar{D}_j(t)}{Q_j}, \forall j \in 1, 2, \ldots M, \tag{24}$$

where Q_j is the degradation tolerance for job type j introduced in Equation (10). We assign $N_{assg}(t)$ idle servers to each cluster based on the order of $w_j(t)$. A large $w_j(t)$ represents an "urgent" scenario in terms of satisfying the SLAs. Thus, clusters with larger $w_j(t)$ are given higher priorities in receiving idle servers. The cluster with the largest $w_j(t)$ first receives its required number of servers $N_{req,j}(t)$, followed by the cluster with the second largest $w_j(t)$, and so on. Clusters with lower priorities are not guaranteed to receive sufficient servers at t to meet with their requirements, but since the QoS feedback $\bar{D}_j(t)$ is updated at every time moment, these clusters would get higher priorities later and receive sufficient servers to meet their SLAs in the long run.

In Action f, it is possible that after satisfying $N_{req,j}(t)$ for every cluster j, some idle servers are still required to be assigned for tracking the RSR signal. In this case, additional idle servers are allocated to clusters in a round robin manner to keep fairness.

D. Regulate the Power and Service Rate of Active Servers. At the beginning of policy execution cycle, we assume all servers run at their maximal service rate by default for the purpose of overall energy conservation and QoS guarantee. However, sometimes to track the signal, servers are required to be slowed down when the signal power cap is low, i.e., in Action d, as decreasing the service rate can reduce the server power.

Since each cluster runs a homogeneous set of jobs, we have assumed that all servers in each cluster j are always running at the same service rate $u_j(t)$ and thus the same power rate $P_j(t)$. We determine $u_j(t)$ by considering both signal tracking and QoS requirements. To guarantee QoS, we set a lower bound of $u_j(t)$ for each cluster j at time t, i.e., $u_{min,j}(t)$, based on Equation (8) and (11), i.e.,

$$u_{min,j}(t) = u_{max,j} \cdot min\left\{1, \frac{\bar{D}_j(t)+1}{\beta Q_j + 1}\right\}. \tag{25}$$

Equation (25) represents that if the current average job system time in cluster j satisfies the constraint in Equation (11), i.e., $\bar{T}_{sys,j}(t) = (\bar{D}_j(t)+1) \cdot T_{min,j} < (\beta Q_j + 1) \cdot T_{min,j}$, then there is room to slow down active servers from the default maximal service rate $u_{max,j}$, otherwise, all active servers should run at their maximal service rate. This lower bound of the service rate also prevents jobs stalling in the server forever.

We then calculate the total maximal possible reduction in data center power consumption at t based on Equation (4) and (25) as

$$P_{max,rd}(t) = \sum_{j=1}^{M} N_j(t) \cdot \alpha_j(u_{max,j} - u_{min,j}(t)). \tag{26}$$

We define a coefficient $\delta(t)$ as

$$\delta(t) = max\left\{1, \frac{P_{con}(t) - P_{tgt}(t+4)}{P_{max,rd}(t)}\right\}, \tag{27}$$

where $P_{con}(t) - P_{tgt}(t+4)$ is the required amount of data center power reduction to perfectly track the signal. Then the service rate $u_j(t)$ for each cluster j is

$$u_j(t) = u_{max,j} - \delta(t) \cdot (u_{max,j} - u_{min,j}(t)). \tag{28}$$

In this way, the power reduction is fairly distributed to clusters based on their lower bounds in service rate.

5 BIDDINGS IN ENERGYQARE

With the proposed EnergyQARE runtime policy introduced in Section 4, this section focuses on designing an EnergyQARE power and reserve bidding strategy that minimizes the data center energy cost. Since we focus on the hour ahead RSR market, the bidding is conducted hourly and the time period for every real-time RSR signal tracking is 1 hour. The energy cost during the hour is calculated as:

$$Cost = \Pi^E \cdot \bar{P} - \Pi^R \cdot R + \Pi^\epsilon \cdot R \cdot \bar{\epsilon}, \tag{29}$$

where Π^E, Π^R, and Π^ϵ are the prices of power, reserve, and penalty on tracking error, respectively. $\bar{\epsilon}$ is the average of the instant signal tracking error along the time frame (i.e., 1 hour). Equation (29) represents that the energy cost of data center in RSR provision equals to the cost on the bid average power consumption $\bar{P}$, reduced by the credit received from bid reserve provision R, and added by the penalty on the statistics (e.g., the mean) of real-time signal tracking error. We formulate an optimization problem to minimize the data center energy cost with additional constraints on tracking error and workload SLAs introduced in Equation (3) and Equation (10), respectively.

We study the sample frequency distributions on the 1-hour trajectories of the tracking error $\epsilon(t)$ and the QoS degradation of jobs in each cluster j (i.e., $D_j(i)$) for a sufficiently large number of simulations. We notice that they fit the Gamma distribution $\Gamma(k,\theta)$ with shape k: $k_\epsilon = \bar{\epsilon}^2/\sigma_\epsilon^2$, $k_{D_j} = \bar{D}_j^2/\sigma_{D_j}^2$ and scale θ: $\theta_\epsilon = \sigma_\epsilon^2/\bar{\epsilon}$, $\theta_{D_j} = \sigma_{D_j}^2/\bar{D}_j$, where $\bar{\epsilon}$, σ_ϵ, $\bar{D}_j$, σ_{D_j} are mean and standard deviation of the tracking error and QoS degradation during the hour. In our experiments, we see that the Gamma distribution fits well with our results with on average less than 3% regression error. Then the optimization problem is formulated as:

$$\begin{array}{lll} \underset{\bar{P}, R, dynamic\ policy}{\text{minimize}} & \Pi^E \cdot \bar{P} - \Pi^R \cdot R + \Pi^\epsilon \cdot R \cdot \bar{\epsilon} & \\ \text{subject to} & \Gamma(k_\epsilon, \theta_\epsilon, \epsilon^{tol}) \geq \eta^\epsilon, & \Gamma(k_{D_j}, \theta_{D_j}, Q_j) \geq \eta_j,\ \forall j, \\ & \bar{P} + R \leq N \cdot P_{Smax}, & \bar{P} - R \geq N \cdot P_{slp}, \\ & \bar{P} \geq 0, & R \geq 0, \end{array} \tag{30}$$

where N is the total number of servers in the data center, P_{Smax} and P_{slp} are the maximal possible server power and power of sleep state, respectively. $N \cdot P_{Smax}$ and $N \cdot P_{slp}$ are maximal and minimal possible power consumption of the data center. We assume the given data center runs the EnergyQARE runtime policy introduced in Section 4 as the "dynamic policy" and solve the bidding $(\bar{P}, R)$ problem accordingly.

Due to the complexity of the problem, instead of applying analytical methods, we solve the optimal solution using numerical methods. We conduct exhaustive search of $(\bar{P}, R)$ over sufficient fine granularity. For each pair of $(\bar{P}, R)$, we first simulate a 1-hour RSR provision period multiple times with varying signals and workload arrival, so as to estimate the mean and standard deviation statistics of $\epsilon(t)$ and $D_j(i)$. Then we apply these statistics to Equation (30) and search for the optimal $(\bar{P}, R)$. An alternative solution is to build regression models of statistics of tracking error and the QoS degradation via $(\bar{P}, R)$ through extensive simulations and leverage these models to solve Equation (30).

6 EXPERIMENTS AND CASE STUDIES

In this section, we simulate data center RSR provision with EnergyQARE in different scenarios and evaluate the signal tracking performance, workload QoS, and the data center monetary savings.

6.1 Methodology

We first simulate workload arrival as a Poisson process (i.e., the arrival time interval follows exponential distribution) with Monte Carlo simulation methods. Evaluation using a real cluster trace is discussed in Section 6.8. In simulation, the workloads are randomly selected from the PARSEC-2.1 benchmark suite. The power-throughput profiles of each job are taken from real-life measurements (in Section 3.2). The workload arrival rate is calculated based on the size of the data center, i.e., the total number of servers N, and the utilization U, i.e., the percentage of servers that are in active state. In this section, by default, we use the data center consisting of 100 servers and at 50% utilization. Results of different data center sizes and utilization are evaluated in case studies. We use the shallow sleep ($T_{tran} = 10s$, $P_{slp} = 10\% P_{Smax}$) as the default sleep state (Gandhi et al. 2012; Isci et al. 2013). We simulate a 3-hour period experiment multiple times with different signal traces and workload arrivals. We extract the middle 1-hour of each 3-hour period for analysis and neglect the warm-up period, as there may initially be a lack of waiting jobs. We evaluate the signal tracking, QoS performance and the energy monetary savings.

6.2 Signal Tracking Performance and Workload QoS

We first conduct experiments on the default settings with multiple random workload traces. In each workload trace, the types of jobs contained, the arrival rate λ_j of each job type j, and the QoS tolerances Q_j in SLAs, are randomly selected. All these traces are constrained to have utilization at $U = 50\%$. To better evaluate the capability of our policy in guaranteeing SLAs, we set some Q_j to small values (shown in Table 2), so that the SLAs are tight and easy to be violated with random policies. In SLAs, η_j is set to 90% for all cases. The tracking error probabilistic constraint is set as $(\epsilon^{tol}, \eta^{\epsilon}) = (0.3, 90\%)$ based on today's market information (PJM 2017c).

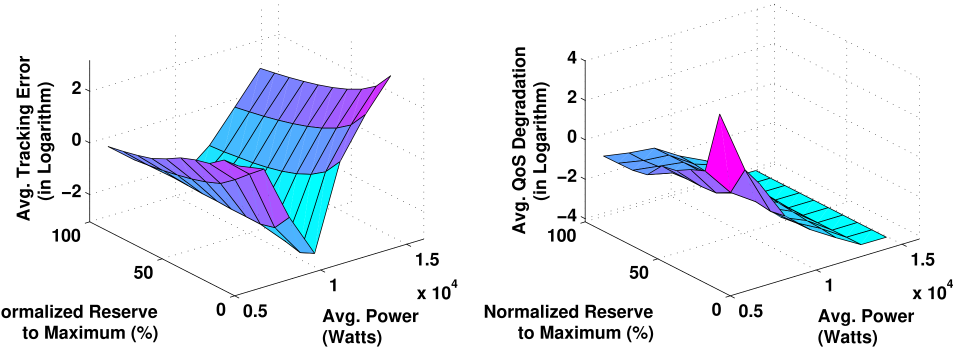
Since results are dependent on workload arrival traces (though the differences are small from our observation), to achieve generalized results, we test on multiple different workload arrival traces, and we randomly select three of them to demonstrate the results. Properties of three selected workload traces are shown in Table 2.

Figure 5 shows the signal tracking performance and the workload QoS via different pairs of bidding values $(\bar{P}, R)$, where R is normalized to its maximal possible value R_{max} given $\bar{P}$, i.e., $R_{max} = min\{N \cdot P_{max} - \bar{P}, \bar{P} - N \cdot P_{slp}\}$. Figure 5(a) is the average tracking error during the 1-hour experiment via $(\bar{P}, R)$ pairs for the workload trace 1. From the figure, tracking error is more sensitive to the average power $\bar{P}$ than the reserve R. The tracking error is large when the average power $\bar{P}$ is either low or high; in other words, the best tracking performance appears in the middle of the range of $\bar{P}$. When $\bar{P}$ is low, the overall power budget to the data center is insufficient to guarantee workload SLAs. As a result, to satisfy the SLAs, the signal power cap is frequently violated, resulting in low signal tracking accuracy. When $\bar{P}$ is high, workloads are fast served and queues are often empty. Then servers are frequently in the idle or sleep states, in which the power cannot be regulated, leading to poor signal tracking performance. Similar results are found in all the other workload traces we have tested.

Figure 5(b) is the average of QoS degradation of the Canneal application in workload trace 1. Unlike the tracking error, the QoS degradation has a monotonic relation to $\bar{P}$: The higher $\bar{P}$ is, the smaller the QoS degradation will be. When $\bar{P}$ is high, the power budget is sufficient to run workloads faster. Similar behaviors are found for all the other applications in trace 1. We also

Table 2. Workloads and Their Properties in Experiments

Job Type j	Shortest Runtime $T_{min,j}$ (s)	Arrival Rate λ_j (# of jobs/s)	QoS Degradation Tolerance Q_j
Workload Trace 1			
Canneal	44.3	0.11	0.3
Bodytrack	51.0	0.10	2.0
Ferret	74.3	0.20	1.0
Freqmine	95.2	0.11	0.1
Facesim	149.6	0.10	0.5
Workload Trace 2			
Blackscholes	23.5	0.43	1.0
Vips	24.5	0.20	0.7
Raytrace	42.4	0.07	0.8
Dedup	58.8	0.14	0.3
Ferret	74.3	0.07	1.0
Fluidanimate	93.3	0.14	2.0
Steamcluster	151.2	0.05	0.5
Workload Trace 3			
Raytrace	42.4	0.12	0.8
Canneal	44.3	0.29	0.3
Bodytrack	51.0	0.25	2.0
Swaptions	74.0	0.07	0.8
X264	93.3	0.11	1.0
Steamcluster	151.2	0.03	0.5

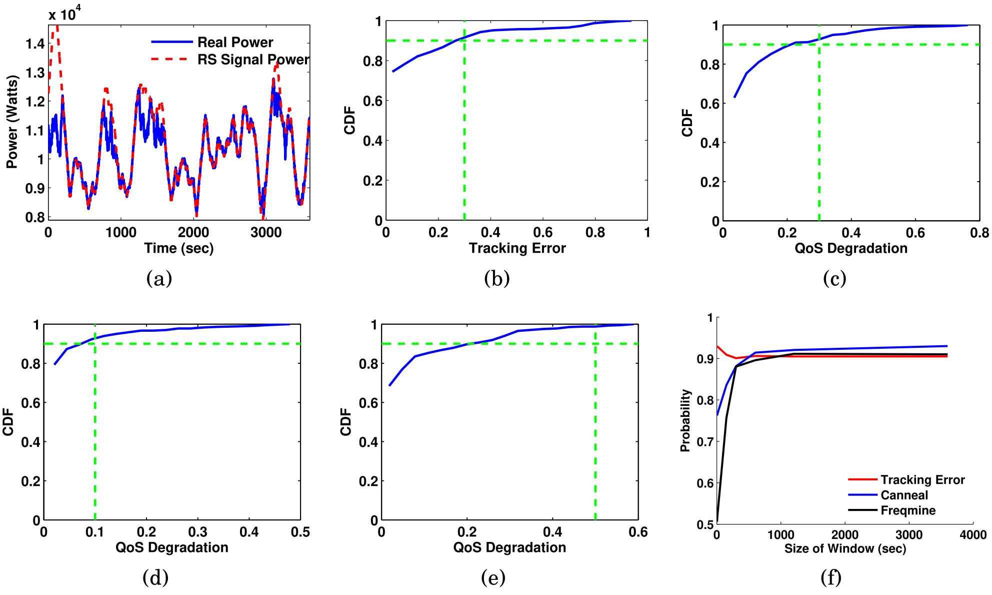


(a) (b)

Fig. 5. The average of the RSR signal tracking error (Figure 5(a)), and the average of QoS degradation of Canneal in workload trace 1 (Figure 5(b)), via different average power $\bar{P}$ and reserve R (normalized to the maximal possible value given $\bar{P}$).

notice that the QoS is more sensitive to R when $\bar{P}$ is low. When $\bar{P}$ is low, a higher R provides larger range in power and more flexibilities in power budgeting and workload servicing and thus leads to better QoS performance. Similar results are found in all the other workload traces we have tested.

Next, we evaluate the signal tracking and the QoS in the runtime given $(\bar{P}, R)$ at the optimal values in Figure 6 (for workload trace 1). Figure 6(a) visualizes the real power consumption $P_{con}(t)$



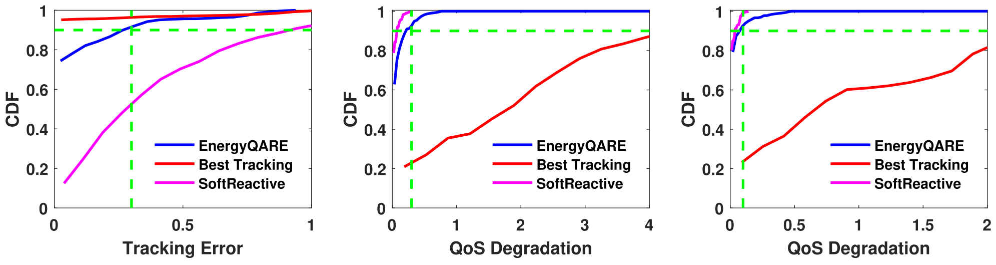
(a) (b) (c) (d) (e) (f)

Fig. 6. Results of RSR signal tracking and QoS degradation for the workload set 1 with $(\bar{P}, R)$ at their optimal values. Figure 6(a) is the real dynamic power consumption $P_{con}(t)$ compared with the RSR signal power cap $P_{tgt}(t)$ during the simulation period. Figure 6(b) is the CDF of the tracking error $\epsilon(t)$. The green lines show the tracking error probabilistic constraints, i.e., $(\epsilon^{tol}, \eta^{\epsilon})$. Figure 6(c)–(e) are the CDF of the QoS degradation of the workloads Canneal, Freqmine, and Facesim, respectively. The green lines are the QoS degradation constraints, i.e., (Q_j, η_j). Figure 6(f) shows the impact of the size of the feedback window T on the tracking error and workload QoS. The probabilities of the tracking error and the QoS degradation that are smaller than ϵ^{tol}, and Q_j for Canneal and Freqmine are shown in the figure.

and the RSR signal power $P_{tgt}(t)$. $P_{tgt}(t)$ is well tracked by $P_{con}(t)$ with only a few notable violations when the signal values are high but there are no sufficient workloads in the system. In fact, the tracking accuracy can be further increased if a lower $\bar{P}$ is selected. However, the workload SLAs are then violated as a side effect. Our solution can better handle the tradeoff between signal tracking and QoS.

Figure 6(b) is the cumulative distribution function (CDF) of the tracking error $\epsilon(t)$ during the 1-hour simulation. The green lines represent the probabilistic constraints $(\epsilon^{tol}, \eta^{\epsilon})$. Figure 6(c)–(e) are the CDF of the QoS degradation of three selected applications from workload trace 1 with the tightest SLA constraints: the Canneal, Freqmine, and Facesim applications (see in Table 2). The green lines represent the SLA probabilistic constraints (Q_j, η_j). Tracking error and workload QoS all meet the constraints from the figures. The rest of two applications (not shown here) in workload trace 1 have even better QoS (i.e., higher probabilities η_j in satisfying the tolerances Q_j) due to the loosen SLA constraints. Moreover, all the other tested traces show similar results to these figures. Overall, our EnergyQARE enables the data center to participate in the RSR provision with accurate signal tracking, while also guaranteeing workload QoS.

Figure 6(f) shows the impact of the feedback window size T (introduced in Equation (5)) on the tracking error and workload QoS. The probabilities of the tracking error and QoS degradation (of Canneal and Freqmine) that are smaller than ϵ^{tol} and Q_j, respectively, are shown in the figure. From the figure, the QoS is poor with a small window size. When the window size is small, the policy makes decisions only based on recent observation of the performance, hence decisions are



(a) CDF of Tracking Error (b) QoS CDF of Canneal (c) QoS CDF of Freqmine

Fig. 7. The CDF of tracking error and QoS degradation of both the EnergyQARE, the best-tracking, and the *SoftReactive* policy for workload trace 1. The blue curves represent the best-tracking policy, and the red curves represent the EnergyQARE. The green lines represent the tracking error and the workload SLA probabilistic constraints, i.e., $(\epsilon^{tol}, \eta^{\epsilon})$ and (Q_j, η_j).

with higher variances. Since the execution time of applications in simulation varies from a few seconds to minutes, short observation in a small window fails to effectively characterize the overall workload QoS and thus leads to inaccurate feedback and poor decisions. In terms of signal tracking, however, the policy with a small window size tends to make decisions based on instant tracking error, which is similar to what the best-tracking policy proposed in a prior study does (Chen et al. 2014a), and therefore, leads to better tracking performance. In addition, Figure 6(f) shows that the tracking performance and workload QoS flatline after the window size increases and reaches a certain value.

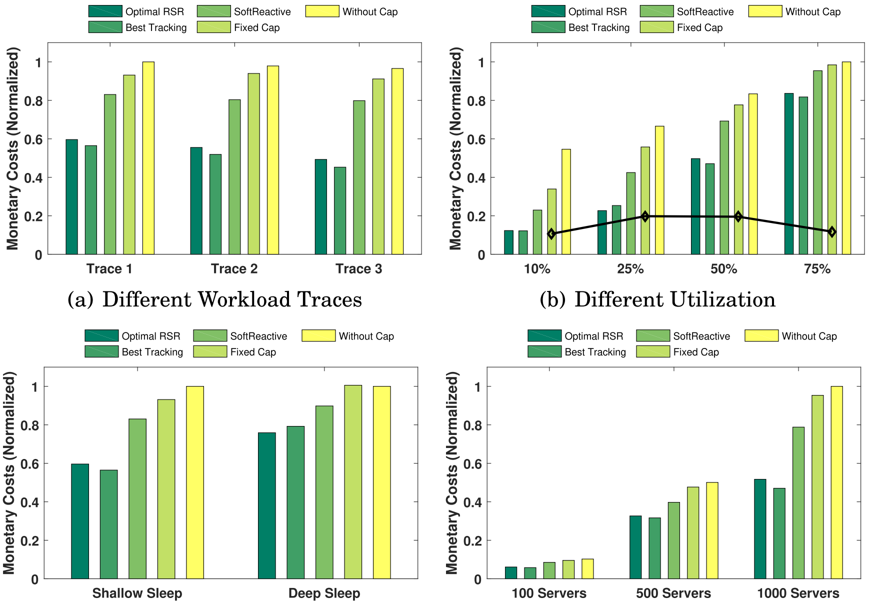
6.3 Comparison of Policies: Tracking Error and Quality of Service

We then compare EnergyQARE with state-of-the-art policies: the best-tracking policy (Chen et al. 2014a) and the *SoftReactive* policy (Gandhi et al. 2012). The best-tracking policy regulates the data center power dynamically to track the RSR signal as accurate as possible, but does not explicitly consider workload QoS. The *SoftReactive* policy simply applies the timeout mechanism to turn servers into sleeping states using the threshold in Equation (20), and it wakes up servers whenever there are more jobs in the queue than the number of idle servers. This policy does not follow the RSR signal.

Figure 7 shows the results of the EnergyQARE, the best-tracking policy, and the *SoftReactive* policy with workload trace 1. The CDF of the tracking error, of the QoS degradation of Canneal and Freqmine are shown in Figure 7(a)–(c), respectively. Green lines represent the probabilitistic constraints on the tracking error and the SLAs, i.e., $(\epsilon^{tol}, \eta^{\epsilon})$ and (Q_j, η_j). From the figure, though the best-tracking policy provides better tracking performance in Figure 7(a), because it minimizes the instantaneous tracking error, the workload QoS of the policy is much worse than that of the EnergyQARE based on Figure 7(b) and (c). The workload SLAs are violated by the best-tracking policy in both figures, due to the fact that the best-tracking policy does not take QoS feedback into account in decision making. The EnergyQARE, however, satisfies all signal tracking and workload SLA constraints. The *SoftReactive* policy is not specifically designed for RSR signal tracking, thus it shows poor tracking performance in Figure 7(a).

6.4 Comparison of Policies: Monetary Savings

In this section, we evaluate the energy monetary savings of data center participation in RSR. In Figure 8(a), we compare the optimal monetary costs of data centers with RSR provision by EnergyQARE, i.e., “optimal RSR,” to the “fixed cap” scenario, the “without cap” scenario, as well as the



(a) Different Workload Traces (b) Different Utilization

(c) Different Sleep States (d) Different Data Center Sizes

Fig. 8. The energy monetary costs in "optimal RSR," "fixed cap," and "without cap" scenarios, as well as the best-tracking and *SoftReactive* policy. All the costs are normalized to the largest value in each figure. In Figure 8(b) we also calculate the absolute value of the *savings* from the "optimal RSR" scenarios to their corresponding "fixed cap" scenarios and represent the savings in the black line.

best-tracking policy (Chen et al. 2014a) and the *SoftReactive* policy (Gandhi et al. 2012) for workload traces 1, 2, and 3. The "fixed cap" scenario represents that the data center consumes the average power at $\bar{P}$ (so the overall energy consumption is close to that of "optimal RSR"), but with no reserve provision, i.e., $R = 0$. The "without cap" scenario represents that the data center serves workloads without any constraints on the power consumption.

The energy monetary costs in "fixed cap," "without cap" scenarios, as well as the *SoftReactive* policy are calculated simply based on the total energy consumed, i.e., $\Pi^E \cdot E$, where E is the total energy consumed during the simulation period (i.e., 1-hour). To better evaluate the savings, we normalize the energy monetary cost of each scenario to the largest value in all the scenarios demonstrated in each figure. As we mentioned in Section 2.2 and 2.3, we make a conservative assumption that the reserve price Π^R is close to the energy price Π^E. Based on today's markets (PJM 2017a, 2017b, 2017c), the energy price Π^E is around 0.1$kWh, and the cost of tracking error is around the reserve price Π^R. Therefore, we use $\Pi^E = \Pi^R = \Pi^\epsilon = 0.1\$/\text{kWh}$ in our formulation.

From Figure 8(a), savings in different workload traces are similar: Our extensive simulations on different traces demonstrate a less than 5% variation in energy monetary savings. Providing RSR with EnergyQARE, the data center saves on average 41% energy cost compared to the "fixed cap" scenario, 44% compared to the "without cap" scenario, and 31% compared to the *SoftReactive* policy, while also satisfying the workload SLAs. To fairly compare "optimal RSR" with best-tracking, when simulating the best-tracking policy, we choose the same $\bar{P}$ and R values as in the "optimal RSR" scenario. Compared to "optimal RSR," the best-tracking policy reduces monetary costs by 5% on average owing to its smaller tracking error. However, as shown in Figure 7(b) and (c), workloads cannot meet their SLAs using the best-tracking policy.

6.5 Case Study 1: Multiple Utilization Settings

We now study how the utilization of data centers impacts the performance and the savings in RSR provision. The utilization of a data center is defined as the average number of active servers. For example, $U = 50\%$ means on average each server is active for half of the whole period and is in idle or sleep state for the rest of the time. The utilization depends on the arrival frequency and the servicing time of the workloads. We simulate the workload arrivals using the same type of applications and the same SLAs to workload trace 1, but with different utilization as 10%, 25%, 50%, and 75%. All information in Table 2 remains the same, except for the arrival rates, which are scaled up for the 75% utilization scenario and scaled down for the 10% and 25% utilization scenarios. The same RSR signal trace is used in all scenarios.

Figure 8(b) shows the data center energy monetary costs of the "optimal RSR," "fixed cap," and "without cap" scenarios, as well as the best-tracking and *SoftReactive* policy in different utilization settings. Costs in all the scenarios and settings are normalized to the maximal value in the figure. For 10%, 25%, 50%, and 75% utilization cases, the savings from the "optimal RSR" to the "fixed cap" scenario are 63.7%, 59.3%, 36.0% and 15.1% and to the "without cap" scenario are 77.4%, 65.9%, 40.4%, and 16.4%, respectively. We see that the percentage of savings increases when the utilization decreases. We also calculate the absolute values of the savings from the "optimal RSR" scenarios to the corresponding "fixed cap" scenarios for all utilization settings and present the results in the black line in Figure 8(b). We notice that the largest absolute value of the monetary saving (i.e., corresponding to the largest amount of reserve provision) appears in the middle level of the utilization. When the utilization is high, the data center is busy with job servicing and has small flexibility in regulating the power and tracking the signal. When the utilization is low, the data center does not have sufficient jobs to be served, which limits the number of servers activated and thus limits the data center power regulation capabilities. Therefore, the amount of RSR provision is small when the data center utilization is either too high or too low.

6.6 Case Study 2: Shallow Sleep vs. Deep Sleep

Many servers in today's data centers support different sleep modes. So far by default we have used a shallow sleep mode that has relatively high sleep power but short transition delay. In this section, we compare shallow sleep mode with a "deep sleep" mode that has longer transition period for waking up, but lower sleep power. In this case study, we use $P_{slp} = 5\% P_{Smax}$, $T_{tran} = 200s$, $P_{tran} = P_{Smax}$ for the deep sleep, and $P_{slp} = 10\% P_{Smax}$, $T_{tran} = 10s$, $P_{tran} = P_{Smax}$ for the shallow sleep, based on recent studies (Gandhi et al. 2012; Isci et al. 2013). We apply workload trace 1 and the same RSR signal trace to both cases for fair comparison.

Figure 8(c) presents the monetary costs using the shallow or the deep sleep mode. Using the shallow sleep, the savings from the "optimal RSR" to the "fixed cap" and the "without cap" are 36.0% and 40.4%, respectively, while using the deep sleep, both numbers drop to 24%. Since RSR provision requires fast and frequent data center power regulation to track the fast-changing signal, the shallow sleep mode is more suitable due to the short delay in state transition and therefore leads to higher savings than the deep sleep mode. We also find that with a deep sleep mode, having a fixed power cap (i.e., "fixed cap") does not provide much energy cost savings compared to "without cap" from the figure. When having a power cap, servers are frequently forced to sleep to meet the cap and woken up later once needed. The deep sleep mode leads to tremendous time delay and energy loss during the frequent switches. As a result, having a power cap with a deep sleep mode is not an efficient strategy in terms of the energy cost savings.

6.7 Case Study 3: Scalability via Data Center Size

Finally, we study the scalability of savings via the size of data centers, i.e., the number of servers in them. We simulate the workload arrivals based on workload trace 1, at 50% utilization with different number of servers. All information in Table 2 remains the same, except for the arrival rates, which are scaled up and down based on the number of servers. Still, we use the same RSR signal trace in simulation. Figure 8(d) shows the monetary costs for data centers containing 100, 500, and 1,000 servers. From the figure, we see similar percentage of savings from RSR provision in all three cases, which demonstrates that the savings are scalable via sizes of data centers.

6.8 Simulations Using a Real Cluster Trace

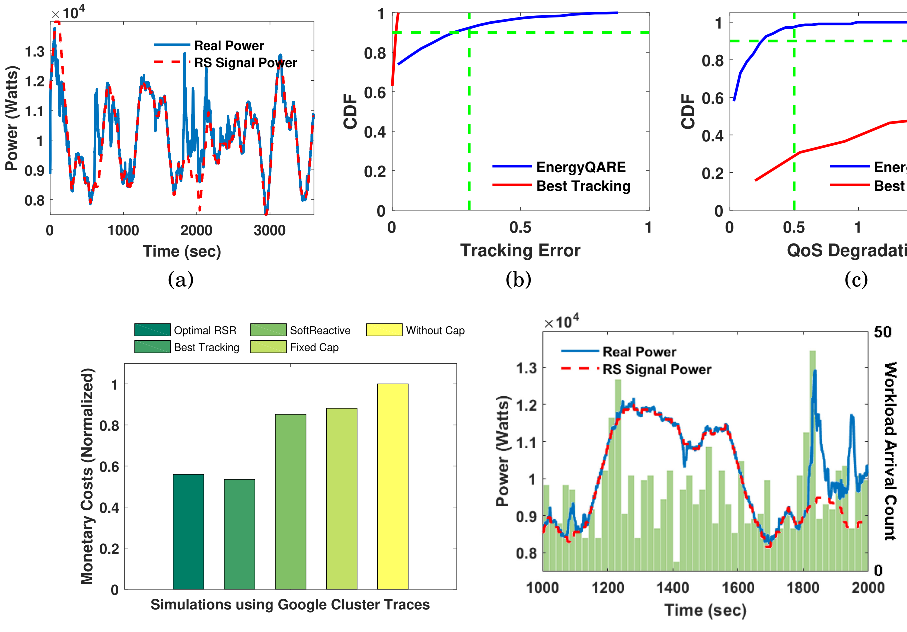
We further evaluate EnergyQARE using a real Google cluster trace (C. Reiss and Hellerstein 2011) as a proxy for workload arrivals. The total length of the trace is one month, and we randomly select days from this trace for our experiments. The following results use the Google trace from May, 18, 2011. During that day, 14,198 jobs are submitted to Google's cluster and executed successfully. Since the trace only contains schedule information and has no power information, we take the arrival time from the Google trace and assume the power usage of these jobs are the same as the 13 benchmarks discussed in Section 3.2. We map the power usage of a benchmark to that of a Google trace job according to the execution time. To be specific, jobs in the Google trace with a relatively smaller execution time are assumed to share the same power usage with a benchmark that also has smaller execution time. To experiment with higher utilizations, we increase the number of jobs in the trace by repeating each arrival multiple times. The QoS constraint (defined in Equation (10)) and the execution time for all 13 benchmarks used in the following simulations are the same as the ones in Table 2.

Figure 9 are results using the Google trace, assuming a data center with 100 servers and at 50% utilization. Figure 9(a) shows that our EnergyQARE enables the data center to follow the RS signal well. Figure 9(b) shows that the tracking error meet the constraint. Figure 9(c) shows the CDF of QoS degradation of Steamcluster. The QoS constraints of other job types are also met well by EnergyQARE. By comparing EnergyQARE with the best-tracking policy, we see in Figure 9(b) that the best-tracking policy follows the signal better than EnergyQARE at the cost of the QoS of workloads, whereas the EnergyQARE policy guarantees that both the tracking error and the QoS meet the constraints. The QoS of other job types when using the best-tracking policy also cannot meet the QoS constraints. Figure 9(d) compares the monetary savings among different policies as we did in Section 6.4. They are quite similar to the results from generated-trace experiments in Section 6.4. We also explore tracking performance and QoS at different $(\bar{P}, R)$ and observe that the results are almost the same as Figure 5.

Figure 9(a) shows that there are temporary noticeable tracking errors. The largest one of them is zoomed in to the time range 1,000~2,000s in Figure 9(e). This tracking error around $t = 1850$ results from a significant variation in workload arrival as shown by the bars in the figure. At $t = 1{,}850$, the workload arrival temporarily increases to be more than twice as large as the average. Since our policy needs to maintain the QoS of the workloads, the real power is unable to match the low RS signal. However, the spike of workload arrival at $t = 1,200$ is handled well, because the signal is higher at that time. Therefore, in real systems that show large variation in workload arrival, we expect some temporary mis-tracking. Nonetheless, Figure 9(b) shows that the overall tracking error meets the constraint on tracking error defined in Equation (3).

7 DISCUSSION

Characteristics of the RSR signal $y(t)$ can be leveraged to potentially improve the current solution. As introduced in Section 2.3, the values of $y(t)$ are not independent from each other, and the



(a) (b) (c)

(d) (e)

Fig. 9. Results from simulations using Google cluster trace. Figure 9(a) is the real power consumption compared with the RSR signal power. Figure 9(b) compares the CDF of the tracking error among different policies. Figure 9(c) compares the CDF of the QoS degradation of workload Steamcluster. Figure 9(d) compares the monetary savings. Figure 9(e) is a zoom-in view of signal tracking in the time range 1,000~2,000s, with the bars showing the number of jobs arriving in each time interval (right axis).

statistical behavior of $y(t)$ can be known ahead. For example, when the signal value is close to its valley, there is a higher chance that the signal will increase rather than decease in the near future. Some previous studies estimate the RSR signal by a Markov chain with two states: the signal value $y(t)$ and the signal change $D(t) = y(t) - y(t-4)$. Then, the signal is estimated as $y(t+4) = f_1(y(t),\ D(t))$, $D(t+4) = f_2(y(t),\ D(t))$. Function $f_1(\cdot)$ and $f_2(\cdot)$ can be calibrated in advance, since the statistic behavior of $y(t)$ is usually known ahead or can be mined from historical data (Zhang et al. 2014).

The current EnergyQARE runtime policy only considers the instant signal value. Better decisions may be made if the policy also predicts future values, or uses statistical information. For example, if there is a high chance that the signal value will increase, consuming slightly more power than the signal to improve the workload QoS may be a better decision than exactly following the signal at that moment.

Not only can the prediction of the RSR signal help improve performance, but also the prediction of workload arrivals can. Predictable workload arrivals may assist the data center to plan ahead. In that case, stochastic methods such as model predictive control or stochastic dynamic programming can be applied to optimize the policy.

In addition, currently we search the optimal $(\bar{P},\ R)$ over a grid with sufficiently fine granularity. Another future direction could be the investigation of alternative search solutions. For example, an sensitivity analysis of the QoS degradation and the signal tracking error via $(\bar{P},\ R)$ may guide for a more structured search.

While this work relies on simulation, our previous experiments on a single server have shown that good tracking performance (with a tracking error reaching up to 5%) and QoS can be achieved by power capping (Chen et al. 2013b). Experimenting with a real cluster is also in our future work agenda.

8 RELATED WORK

This section first reviews data center power management methods, followed by a literature review on data center participation in both legacy and emerging DR programs.

8.1 Server and Data Center Power Management

Server dynamic power management has been widely studied. Dynamic voltage-frequency scaling (DVFS) manages server power by leveraging different CPU frequency settings (Li and Martinez 2006; Reda et al. 2012). Modern servers support putting idle units into intermediate states instead of simply turning them off, which reduces transition cost (Meisner et al. 2009; Isci et al. 2013). For multi-core processors and multi-threaded applications, threads packing, as well as their dynamic allocation and migration offer additional degrees of freedom in power management (Teodorescu and Torrellas 2008; Rangan et al. 2009). Voltage and frequency islands (VFIs) are introduced to achieve fine-grained system level power management (Ogras et al. 2007). Due to the increasing demand of limiting power, many studies focus on power capping methods (David et al. 2010; Ma and Wang 2012). Resource management and workload consolidation are employed in modulating the power of virtual machines (VMs) (Hwang et al. 2012; Dhiman et al. 2009). CPU resource limits are investigated for VMs that can provide finer granularity than DVFS (Chen et al. 2013b).

In addition to single server power management, there are data center level power management techniques, such as power budgeting and server provisioning. Power budgeting determines how to budget the total data center power to each server and cooling unit (Zhan and Reda 2013; Rajamani et al. 2006). Server provisioning determines how many servers should be activated to serve workload at a given time and which states other servers should be put in. Traditional data centers seldom sleep or turn off idle servers to guarantee workload QoS conservatively. Being aware of the tremendous energy waste, recent studies investigate putting unused servers into power saving states, mainly the sleeping state, though most of them use simple models and hypothetical server states that ignore the transition costs during state switches (Chase et al. 2001; Meisner et al. 2009). Recently, Gandhi et al. introduce a timeout mechanism to smartly put servers into sleep in their *SoftReactive* policy (Gandhi et al. 2012).

8.2 Data Center Demand Response

Data center participation in DR programs has been significantly advanced in recent years. The opportunities and challenges of data center in both legacy and emerging DR programs are surveyed (Wierman et al. 2014; Kirpes and Klingert 2016). Real-time dynamic energy pricing (Le et al. 2016), peak shaving (Wang et al. 2012; Govindan et al. 2011) and emergency load reduction (Zhang et al. 2015; Tran et al. 2016; Islam et al. 2016) are three most popular legacy DR programs. Data centers participate in legacy DR programs mostly through load shedding, shifting and migration (Ghamkhari and Mohsenian-Rad 2012; Liu et al. 2014; Wang et al. 2014; Cioara et al. 2018). Load shedding reduces load by turning off servers, without any future pay back. Load shifting temporarily turns off servers but reschedules loads to a future spot (Ghatikar et al. 2012). Load migration shifts load geographically to other data centers (Wang et al. 2014). Unlike load shedding and shifting, load migration causes less or even no QoS degradation in workload servicing. Some migration strategies have been proposed, and the environmental benefits are evaluated (Chiu et al. 2012; Hu et al. 2016; Wang et al. 2013). Apart from load management, enabling data center

DR participation with ancillary systems, such as ESDs (Urgaonkar et al. 2011; Wang et al. 2012; Kim et al. 2014), or PHEVs (Cui et al. 2015; Li et al. 2014) is another popular research direction.

There is a growing interest in data center participation in emerging DR programs, i.e., capacity reserves. Aikema et al. review multiple types of ancillary service markets for data centers to participate in and evaluate the capability and benefit (Aikema et al. 2012). A systematic comparison demonstrates that RSR is more suitable and profitable for data centers than legacy DR programs and other types of capacity reserves (Chen et al. 2014b). A few control policies for data centers RSR provision have been proposed (Chen et al. 2013b; Ghasemi-Gol et al. 2014; Aksanli and Rosing 2014; Chen et al. 2015b). Some work also studies the joint management of data center and employee PHEVs in RSR provision (Li et al. 2014). However, existing studies use highly simplified data center models for RSR provision, while none of them use practical models that consider heterogeneous workloads, different server power states, transition costs, and workload SLAs. Moreover, none of them study the optimal power and reserve provisioning problem together with the design of online runtime policies.

Our earlier work proposes an online policy that regulates data center power to track the RSR signal as accurate as possible, however, without considering any workload QoS constraints (Chen et al. 2014a). To the best of our knowledge, this work is the first to address all these aspects together in data center RSR provision. We propose practical models for workloads, servers and data centers. Our designed EnergyQARE runtime policy and the bidding mechanism consider both signal tracking and workload QoS. The policy uses both signal tracking and workload QoS as feedback in decision making and leverages server-level power management, server provisioning, power budgeting and workload control together. The bidding strategy solves the optimal power and reserve provisioning problem along with the runtime policy.

9 CONCLUSION AND FUTURE WORK

In this article, we propose EnergyQARE, a novel system that enables data centers to participate in RSR provision in practical scenarios. In EnergyQARE, a practical data center model is first introduced, which takes heterogeneous workloads, different server power states, and workload SLAs into account. Then, we propose an EnergyQARE runtime policy that dynamically regulates data center power following the RSR signal, while also guaranteeing workload SLAs. The policy utilizes historical feedback on both signal tracking error and workload QoS in decision making. It leverages not only server power management but also server provisioning techniques. We also design an optimal energy and reserve bidding strategy to minimize the overall data center electricity cost. We simulate the RSR provision in multiple scenarios with different workload arrivals, server power states, utilization, and sizes of data centers. Results show that in a general data center scenario with 50% utilization, providing RSRs with EnergyQARE offers the data center an average of 44% energy monetary savings, while also guaranteeing tight workload SLAs. The amount of savings is not sensitive to job types, but is highly correlated with the data center utilization and parameters in server power states. The savings are scalable via sizes of data centers. In the future, one research direction could be investigating stochastic methods to figure out optimal solutions when the RSR signal and workload arrivals are predictable, or the statistical information of them is available. Investigating data center RSR provision associated with energy storage devices is another interesting direction.

REFERENCES

D. Aikema, R. Simmonds, and H. Zareipour. 2012. Data centres in the ancillary services market. In *Proceedings of the International Green Computing Conference (IGCC'12)*. 1–10.

B. Aksanli and T. Rosing. 2014. Providing regulation services and managing data center peak power budgets. In *Proceedings of the Conference on Design, Automation & Test in Europe (DATE'14)*. 143.

Amazon. 2013. Amazon EC2 Service Level Agreement. Retrieved from https://aws.amazon.com/ec2/sla.

AWEA. 2015. 2015 U.S. Wind Industry Market Reports. Retrieved from http://www.awea.org/Advocacy.

E. Bilgin, M. C. Caramanis, I. C. Paschalidis, and C. G. Cassandras. 2016. Provision of regulation service by smart buildings. *IEEE Trans. Smart Grid* 7, 3 (May 2016), 1683–1693.

C. Bohringer, A. Loschel, U. Moslener, and T. F. Rutherford. 2009. EU climate policy up to 2020: An economic impact assessment. *Energ. Econ.* 31, 2 (2009), S295–S305.

J. Wilkes C. Reiss and J. L. Hellerstein. 2011. *Google Cluster-usage Traces: Format + Schema*. Technical Report. Google Inc., Mountain View, CA.

J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat, and R. P. Doyle. 2001. Managing energy and server resources in hosting centers. *SIGOPS Oper. Syst. Rev.* 35, 5 (Oct. 2001), 103–116.

H. Chen, M. C. Caramanis, and A. K. Coskun. 2014a. The data center as a grid load stabilizer. In *Proceedings of the 19th Asia and South Pacific and Design Automation Conference (ASP-DAC'14)*. 105–112.

H. Chen, M. C. Caramanis, and A. K. Coskun. 2014b. Reducing the data center electricity costs through participation in smart grid programs. In *Proceedings of the International Green Computing Conference (IGCC'14)*. 1–10.

H. Chen, A. K. Coskun, and M. C. Caramanis. 2013a. Real-time power control of data centers for providing regulation service. In *Proceedings of the IEEE Conference on Decision and Control (CDC'13)*. 4314–4321.

H. Chen, C. Hankendi, M. C. Caramanis, and A. K. Coskun. 2013b. Dynamic server power capping for enabling data center participation in power markets. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD'13)*.

H. Chen, Z. Liu, A. K. Coskun, and A. Wierman. 2015a. Optimizing energy storage participation in emerging power markets. In *Proceedings of the International Green Sustainable Computing Conference (IGSC'15)*. 1–6.

H. Chen, B. Zhang, M. C. Caramanis, and A. K. Coskun. 2015b. Data center optimal regulation service reserve provision with explicit modeling of quality of service dynamics. In *Proceedings of the IEEE Conference on Decision and Control (CDC'15)*. 7207–7213.

D. Chiu, C. Stewart, and B. McManus. 2012. Electric grid balancing through low cost workload migration. *SIGMETRICS Perf. Eval. Rev.* 40, 3 (2012), 48–52.

B. Christian. 2011. *Benchmarking Modern Multiprocessors*. Ph.D. Dissertation. Princeton University.

T. Cioara, I. Anghel, M. Bertoncini, I. Salomie, D. Arnone, M. Mammina, T. Velivassaki, and M. Antal. 2018. Optimized flexibility management enacting data centres participation in smart demand response programs. *Future Generation Computer Systems* 78 (2018), 330–342.

T. Cui, Y. Wang, S. Chen, Q. Zhu, S. Nazarian, and M. Pedram. 2015. Optimal control of PEVs for energy cost minimization and frequency regulation in the smart grid accounting for battery state-of-health degradation. In *Proceedings of the 52nd Design Automation Conference (DAC'15)*. 134:1–134:6.

H. David, E. Gorbatov, U. R. Hanebutte, R. Khanna, and C. Le. 2010. RAPL: Memory power estimation and capping. In *Proceedings of the 2010 ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED'10)*. 189–194.

M. Dayarathna, Y. Wen, and R. Fan. 2016. Data center energy consumption modeling: A survey. *IEEE Commun. Surv. Tutor.* 18, 1 (2016), 732–794.

G. Dhiman, G. Marchetti, and T. Rosing. 2009. vGreen: A system for energy efficient computing in virtualized environments. In *Proceedings of the ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED'09)*. ACM, 243–248.

EIA. 2014. Annual Energy Outlook 2014. Retrieved from http://www.eia.gov/forecasts/aeo.

D. Fooladivanda, C. Rosenberg, and S. Garg. 2014. An analysis of energy storage and regulation. In *Proceedings of the 2014 IEEE International Conference on Smart Grid Communications (SmartGridComm'14)*. 91–96.

A. Gandhi, M. Harchol-Balter, and M. A. Kozuch. 2012. Are sleep states effective in data centers? In *Proceedings of the International Green Computing Conference (IGCC'12)*. 1–10.

M. Ghamkhari and H. Mohsenian-Rad. 2012. Data centers to offer ancillary services. In *Proceedings of the IEEE International Conference on Smart Grid Communications (SmartGridComm'12)*. 436–441.

M. Ghasemi-Gol, Y. Wang, and M. Pedram. 2014. An optimization framework for data centers to minimize electric bill under day-ahead dynamic energy prices while providing regulation services. In *Proceedings of the International Green Computing Conference (IGCC'14)*. 1–9.

G. Ghatikar, V. Ganti, N. Matson, and M. A. Piette. 2012. *Demand Response Opportunities and Enabling Technologies for Data Centers: Findings from Field Studies*. Technical Report. LBNL.

Í. Goiri, M. E. Haque, K. Le, R. Beauchea, T. D. Nguyen, J. Guitart, J. Torres, and R. Bianchini. 2015. Matching renewable energy supply and demand in green datacenters. *Ad Hoc Netw.* 25, Part B (2015), 520–534.

S. Govindan, A. Sivasubramaniam, and B. Urgaonkar. 2011. Benefits and limitations of tapping into stored energy for datacenters. In *Proceedings of the 38th International Symposium on Computer Architecture (ISCA'11)*. ACM, New York, NY, 341–352.

J. Hansen, J. Knudsen, and A. M. Annaswamy. 2014. Demand response in smart grids: Participants, challenges, and a taxonomy. In *Proceedings of the IEEE Conference on Decision and Control (CDC'14)*. 4045–4052.

H. Hu, Y. Wen, L. Yin, and L. Qiu. 2016. Towards cost-efficient workload scheduling for a tango between geo-distributed data center and power grid. In *Proceedings of the International Conference on Communications (ICC'16)*. 1–7.

I. Hwang, T. Kam, and M. Pedram. 2012. A study of the effectiveness of CPU consolidation in a virtualized multi-core server system. In *Proceedings of the ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED'12)*. ACM, 339–344.

C. Isci, S. McIntosh, J. Kephart, et al. 2013. Agile, efficient virtualization power management with low-latency server power states. In *Proceedings of the International Symposium on Computer Architecture (ISCA'13)*. ACM, 96–107.

M. A. Islam, X. Ren, S. Ren, A. Wierman, and X. Wang. 2016. A market approach for handling power emergencies in multi-tenant data center. In *Proceedings of the IEEE International Symposium on High Performance Computer Architecture (HPCA'16)*. 432–443.

Y. Kim, V. Raghunathan, and A. Raghunathan. 2014. Design and management of hybrid electrical energy storage systems for regulation services. In *Proceedings of the International Green Computing Conference (IGCC'14)*. 1–9.

B. Kirpes and S. Klingert. 2016. Evaluation process of demand response compensation models for data centers. In *Proceedings of the Workshop on Energy Efficient Data Centres (E2DC'16)*. 4:1–4:6.

T. N. Le, Z. Liu, Y. Chen, and C. Bash. 2016. Joint capacity planning and operational management for sustainable data centers and demand response. In *Proceedings of the 7th International Conference on Future Energy Systems (e-Energy'16)*. 16:1–16:12.

A. Leon-Garcial. 2008. *Probability, Statistics, and Random Processes for Electrical Engineering*. Prentice Hall.

J. Li and J. F. Martinez. 2006. Dynamic power-performance adaptation of parallel computation on chip multiprocessors. In *Proceedings of the IEEE International Symposium on High Performance Computer Architecture (HPCA'06)*. 77–87.

S. Li, M. Brocanelli, W. Zhang, and X. Wang. 2014. Integrated power management of data centers and electric vehicles for energy and regulation market participation. *IEEE Trans. Smart Grid* 5, 5 (Sep. 2014), 2283–2294.

Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang, M. Marwah, and C. Hyser. 2012. Renewable and cooling aware workload management for sustainable data centers. *SIGMETRICS Perform. Eval. Rev.* 40, 1 (2012), 175–186.

Z. Liu, I. Liu, S. Low, and A. Wierman. 2014. Pricing data center demand response. *SIGMETRICS Perform. Eval. Rev.* 42, 1 (Jun. 2014), 111–123.

K. Ma and X. Wang. 2012. PGCapping: Exploiting power gating for power capping and core lifetime balancing in CMPs. In *Proceedings of the 21st International Conference on Parallel Architectures and Compilation Techniques (PACT'12)*. ACM, New York, NY, 13–22.

D. Meisner, B. T. Gold, and T. F. Wenisch. 2009. PowerNap: Eliminating server idle power. *ACM Sigplan Not.* 44, 3 (2009), 205–216.

NYISO. 2013. *Manual 2: Ancillary Services Manual, v3.26*. Manual. NYISO.

U. Y. Ogras, R. Marculescu, P. Choudhary, and D. Marculescu. 2007. Voltage-frequency island partitioning for GALS-based networks-on-chip. In *Proceedings of the Design Automation Conference (DAC'07)*. IEEE, 110–115.

A. L. Ott. 2003. Experience with PJM market operation, system design, and implementation. *IEEE Trans. Power Syst.* 18, 2 (2003), 528–534.

PJM. 2005. *Integrating Demand and Response into the PJM Ancillary Service Markets*. White Paper. PJM.

PJM. 2017a. Apr 2017 PJM reserve LMP file: www.pjm.com/pub/account/pjm-regulation-data/201704.csv. April 1st, 2017 PJM Day-ahead energy LMP file: www.pjm.com/pub/account/lmpda/20170401-da.zip. April 2nd, 2017 file: www.pjm.com/pub/account/lmpda/20170402-da.zip. April 3rd, 2017 file: www.pjm.com/pub/account/lmpda/20170403-da.zip.

PJM. 2017b. *PJM Manual 11: Energy & Ancillary Services Market Operations*. Manual. PJM. Retrieved from http://www.pjm.com/ /media/documents/manuals/m11.ashx.

PJM. 2017c. *PJM Manual 12: Balancing Operations*. Manual. PJM. Retrieved from http://www.pjm.com/ /media/documents/manuals/m12.ashx.

K. Rajamani, H. Hanson, J. Rubio, S. Ghiasi, and F. Rawson. 2006. Application-aware power management. In *Proceedings of the IEEE International Symposium on Workload Characterization*. IEEE, 39–48.

K. K. Rangan, G. Wei, and D. Brooks. 2009. Thread motion: Fine-grained power management for multi-core systems. *SIGARCH Comput. Archit. News* 37, 3 (Jun. 2009), 302–313.

S. Reda, R. Cochran, and A. K. Coskun. 2012. Adaptive power capping for servers with multithreaded workloads. *IEEE Micro* 32, 5 (Sep. 2012), 64–75.

A. Shehabi, S. Smith, N. Horner, I. Azevedo, R. Brown, J. Koomey, E. Masanet, D. Sartor, M. Herrlin, and W. Lintner. 2016. United states data center energy usage report. *Lawrence Berkeley National Laboratory, Berkeley, California. LBNL-1005775 Page* 4 (2016).

R. Teodorescu and J. Torrellas. 2008. Variation-aware application scheduling and power management for chip multiprocessors. *ACM SIGARCH Comput. Arch. News* 36, 3 (2008), 363–374.

N. H. Tran, C. Pham, S. Ren, Z. Han, and C. S. Hong. 2016. Coordinated power reduction in multi-tenant colocation datacenter: An emergency demand response study. In *Proceedings of the International Conference on Communications (ICC'16)*. 1–6.

O. Tuncer, K. Vaidyanathan, K. Gross, and A. K. Coskun. 2014. CoolBudget: Data center power budgeting with workload and cooling asymmetry awareness. In *Proceedings of the 2014 IEEE 32nd International Conference on Computer Design (ICCD'14)*. 497–500.

R. Urgaonkar, B. Urgaonkar, M. J. Neely, and A. Sivasubramaniam. 2011. Optimal power cost management using stored energy in data centers. In *Proceedings of the ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems*. 221–232.

A. Verma, L. Pedrosa, M. Korupolu, D. Oppenheimer, E. Tune, and J. Wilkes. 2015. Large-scale cluster management at google with borg. In *Proceedings of the 10th European Conference on Computer Systems (EuroSys'15)*. 18:1–18:17.

D. Wang, C. Ren, A. Sivasubramaniam, B. Urgaonkar, and H. Fathy. 2012. Energy storage in datacenters: What, where, and how much? *ACM SIGMETRICS Perf. Eval. Rev.* 40, 1 (2012), 187–198.

H. Wang, J. Huang, X. Lin, and H. Mohsenian-Rad. 2014. Exploring smart grid and data center interactions for electric power load balancing. *ACM SIGMETRICS Perf. Eval. Rev.* 41, 3 (2014), 89–94.

R. Wang, N. Kandasamy, C. Nwankpa, and D. R. Kaeli. 2013. Data centers as controllable load resources in the electricity market. In *Proceedings of the International Conference on Distributed Computing Systems.*

T. Wei, Q. Zhu, and M. Maasoumy. 2014. Co-scheduling of HVAC control, EV charging and battery usage for building energy efficiency. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD'14)*. 191–196.

A. Wierman, Z. Liu, I. Liu, and H. Mohsenian-Rad. 2014. Opportunities and challenges for data center demand response. In *Proceedings of the International Green Computing Conference (IGCC'14)*. 1–10.

C. Yang, A. Wierman, S. Shakkottai, and M. Harchol-Balter. 2012. Many flows asymptotics for SMART scheduling policies. *IEEE Trans. Automat. Control* 57, 2 (Feb. 2012), 376–391.

X. Zhan and S. Reda. 2013. Techniques for energy-efficient power budgeting in data centers. In *Proceedings of the Design Automation Conference (DAC'13)*. 176.

B. Zhang, M. C. Caramanis, and J. Baillieul. 2014. Optimal price-controlled demand response with explicit modeling of consumer preference dynamics. In *Proceedings of the IEEE Conference on Decision and Control (CDC'14)*. 2481–2486.

L. Zhang, S. Ren, C. Wu, and Z. Li. 2015. A truthful incentive mechanism for emergency demand response in colocation data centers. In *Proceedings of the 2015 IEEE Conference on Computer Communications (INFOCOM'15)*. 2632–2640.

D. Zhu, Y. Wang, S. Yue, Q. Xie, M. Pedram, and N. Chang. 2013. Maximizing return on investment of a grid-connected hybrid electrical energy storage system. In *Proceedings of the Design Automation Conference, 19th Asia and South Pacific (ASP-DAC'13)*. 638–643.

Received August 2016; revised June 2018; accepted July 2018