Data Center and Load Aggregator Coordination Towards Electricity Demand Response

Yijia Zhang[1,2]*, Athanasios Tsiligkaridis[1]*, Ioannis Ch. Paschalidis[1], and Ayse K. Coskun[1]†

[1]Department of Electrical and Computer Engineering, Boston University, Boston, MA, USA 02215

[2]Peng Cheng Laboratory, Shenzhen, China

*These authors contributed equally.

†Correspondence: acoskun@bu.edu

ABSTRACT

In a demand response scenario, coordinating multiple data centers with an electricity load aggregator provides opportunities to minimize electricity cost and absorb the volatility in the grid that is caused by renewable generation. To enable optimal coordination, this paper introduces a joint data center and aggregator optimization framework that minimizes the cost of data centers while they participate in demand response programs regulated by a load aggregator. The proposed framework, *DCAopt*, solves three integrated optimization problems: optimizing the quality-of-service of jobs in each data center, coordinating workload sharing among multiple data centers, and assigning (electricity) prices that incentivize demand response. Instead of relying on simplified relations between a data center's overall utilization rate and the average job delay, DCAopt applies queueing theory and job scheduling simulation techniques to model data centers with heterogeneous workloads, where different workload properties can be measured using data from actual servers. DCAopt solves the aforementioned joint optimization problems via gradient descent. Through evaluation using fine-grained simulations, we demonstrate that our framework finds better solutions to the data-center-aggregator optimization problems. With DCAopt, the energy costs of data centers can be reduced by 5% on average, with a corresponding reduction of a social cost assessed by the aggregator amounting to more than 30% in most cases. In addition, power usage reduction at the data centers is 6% higher compared to data-center-centric power use optimization.

1. Introduction

Generation of renewable energy supply is expected to grow substantially in the coming decades to help battle climate change. However, the intermittency of renewable generation caused by the volatility of weather conditions poses a serious challenge on the matching of supply and demand in the electricity grid, mainly necessitated by the lack of large-scale energy storage. To address this challenge in the era of renewable energy, power consumers are encouraged to participate in *demand response programs,* which motivate the demand side (i.e., the power consumers) to regulate their power consumption following market requirements.

Data centers are essential parts of our societal and economic fabric as they offer computing resources to support businesses, government operations, consumer applications, and scientific research. While data centers provide computing capabilities as large as hundreds of peta floating-point operations per second (petaflops), they are also tremendous power consumers. In 2014, data centers in the United States consumed 70 TWh, representing about 1.8% of total US electricity consumption [1]. In 2018, the global data center energy usage was estimated to be 205 TWh, corresponding to around 1% of global electricity consumption [2], with that share expected to increase as more data centers are currently under construction. Large power consumption inevitably leads to large electricity costs, and large companies, such as Google, spend hundreds of millions of dollars on electricity costs every year. On the other side, data centers are capable of regulating their power consumption in large ranges through control "knobs" including job scheduling and dynamic voltage-frequency scaling [3]. As a result, data centers are potentially important candidates for participation in demand response programs.

In a demand response scenario, when handling a supply deficit or a demand surplus event, a load aggregator can regulate electricity prices and use them as incentives to motivate data centers to meet a total load reduction target. In addition, coordinating the workload and power consumption of multiple data centers together offers an opportunity to further reduce the total electricity cost. For example, when multiple data centers are in different locations and face different electricity prices, they have an incentive to share workload from one location to another. Assuming there are two data centers at two locations, when the electricity price at the location of data center 1 is lower than that of data center 2, data center 1 could allocate some of its servers to run jobs submitted to data center 2, and the total electricity cost of the two data centers can be correspondingly reduced. Essentially, the communication infrastructure used to exchange jobs can also be used to balance the electric grid.

Recent research has evaluated the benefits of employing data centers to participate in demand response programs as well as coordinating multiple data centers in a smart grid [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20]. However, the relevant smart-grid literature does not focus on the special aspects of data centers but instead views them as loads [11] [14] [16] [18] [20], which leads to data center models that are based on lumped parameters, such as the total workload arrival rate, the average server utilization, etc. These lumped parameters ignore the heterogeneity of workloads and the complexity of the job scheduling process in data centers, which hinders the applicability of such approaches to practical data center scenarios. Meanwhile, other data center-related literature considering heterogeneous workloads or a concrete job scheduling process, mostly focuses on optimization problems from the data center side given demand response signals from the smart grid [10] [12] [17] [19]; these works do not consider the bidirectional interactions between the electricity grid and data centers in a joint optimization as we do in this paper.

To fill this gap, in this work we introduce a joint *Data Center and Aggregator optimization framework*, *DCAopt*, that captures sufficient details of both systems and considers their interactions. DCAopt finds the optimal strategy for data centers to participate in demand response and for the aggregator to achieve power reduction. On the one side,

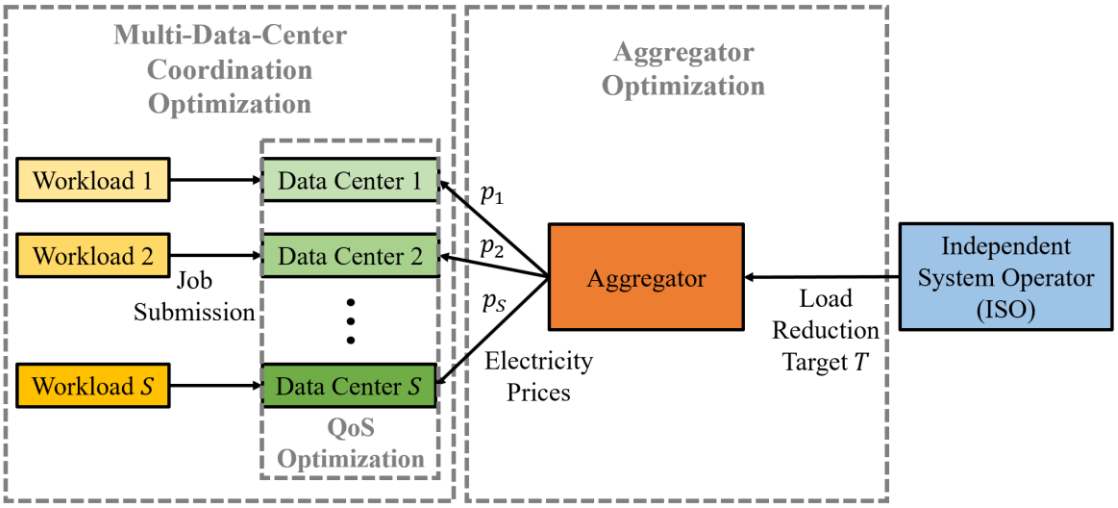

Figure 1. The interactions between data centers, the aggregator, and the ISO in our DCAopt framework. DCAopt simultaneously solves three optimization problems: (1) Each data center solves a QoS optimization problem to maximize the QoS of jobs being processed in that data center. (2) All data centers together solve a multi-data-center coordination optimization problem to minimize the total electricity cost by applying workload sharing and optimizing server arrangement. (3) The aggregator solves a social cost minimization problem to achieve the load reduction target from the ISO by assigning optimized electricity prices to the data centers.

DCAopt enables the aggregator to handle supply deficit events, and on the other side, DCAopt enables data centers to minimize their costs by coordinating workload and power consumption through workload sharing. Our work employs a data center model implemented in a simulation environment, accepting a heterogenous workload composed of jobs with different properties (job arrival rate, typical execution time, waiting time constraint) to fill multiple queues, and capturing job events (submit / start / finish) on the timescale of seconds. Our work also takes the quality-of-service (QoS) of data center jobs into consideration while performing the optimization. Through evaluation using fine-grained simulations, we demonstrate that our framework finds better solutions to the data-center-aggregator optimization problems. With DCAopt, the energy costs of data centers can be reduced by 5% on average, with a corresponding reduction of a social cost assessed by the aggregator amounting to more than 30% in most cases. In addition, power usage reduction at the data centers is 6% higher compared to data-center-centric power use optimization.

In summary, this work makes the following contributions:

- We propose a joint data center and aggregator optimization framework, DCAopt, which provides an optimal strategy for data centers to participate in demand response programs, while also maximizing benefits for the load aggregator on the electricity grid side.
- DCAopt applies both queueing theory and job scheduling simulation techniques to optimize the Quality-of-Service of data centers, which accept heterogeneous workloads composed of jobs with different properties (job arrival rate, job execution time, time constraint) to fill multiple queues.
- We conduct simulations with workload properties measured from actual operational servers and compare the benefits of DCAopt under several different scenarios. We show that DCAopt can reduce the energy costs of data centers by 5% on average and can also reduce the social cost by more than 30% in most cases.

2. Related work

Resource allocation in data centers is a challenging topic due to the heterogeneity of systems/workloads and various operational constraints (including QoS and resource availability) that add significant complexity to the resource allocation problem. Many strategies to improve data center resource allocation have been proposed recently [21] [22] [23]. One of the central goals of optimizing data center's resource allocation is to reduce the power consumption and electricity bill while still being able to finish computing tasks on time. Many software methods to improve the power efficiency of data centers have been proposed in recent years [24], including processor dynamic frequency-voltage scaling (DVFS) [3], thread packing [25], uncore frequency adaptation [26], server power-saving states [27], hardware over-provisioning [28], and on/off network links [29].

Based on those methods, especially the processor DVFS and the server power-saving states, data centers are able to participate in demand response and reduce their electricity consumption costs [10] [12] [17] [19]. For example, by considering the dynamic electricity price and the fluctuation of renewable energy supply, Lei et al. proposed an energy-efficient job scheduling strategy to modulate a data center's total power, and they showed that the electricity bill of the data center can be reduced significantly [8]. Jahanshahi et al. used a detailed data center model considering the different power states of CPUs and proposed a power management approach for data centers to participate in the frequency response regulation service of the smart grid [19]. However, these works discussed above only focus on how a data center reacts to the requirements from the market, and they do not consider the bidirectional interactions between the electricity grid and the data centers.

To utilize the regional difference in the electricity prices, resource availability, and workload characteristics of data centers, the potential of multi-data-center coordination through workload migration has been investigated in a number of works [5] [6] [7] [30] [10] [11] [18]. For example, Lin et al. proposed the use of a receding horizon control algorithm to optimize the load balancing of data centers at different locations [6]. Their works also developed algorithms for coordinating local electricity generation with workload shifting among data centers [5] [7]. Niu et al. applied the model of stochastic economic dispatch to solve the spatial coordination between data centers and the power system [18]. Some of these works also discussed the mutual interaction

between the grid and data centers [11] [14] [16]. For example, Zhou et al. proposed an auction mechanism between smart grids and geo-distributed data centers for maximizing social welfare [11].

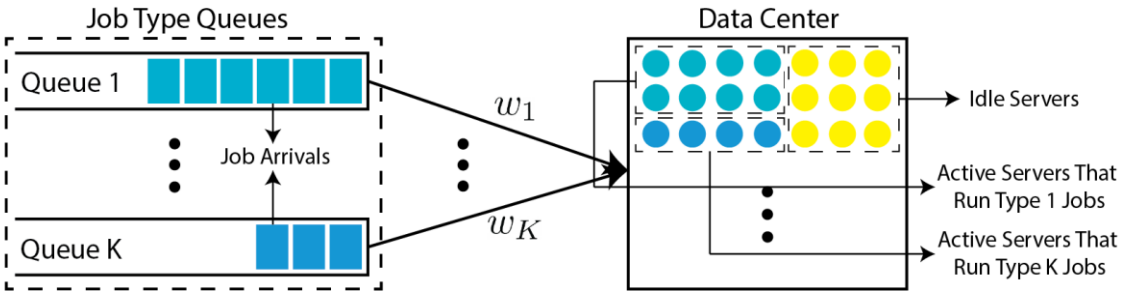

Figure 2. The model of a data center with separate queues serving different types of jobs. Servers switch between the idle state and the active state as they start or finish a job. We apply the generalized processor sharing algorithm with weights w_k $(k = 1,2, \dots)$ to control the number of servers in each active server group.

However, although these works have considered the bidirectional interactions between the electricity grid and the data centers, they usually model the performance and power of data centers using lumped parameters such as the total workload arrival rate and the average server utilization. For example, Niu et al. assumed that the workload assigned to a data center at a given time is parameterized by a single scalar value [18], and their approach adjusts the data center utilization rate l to meet a QoS constraint in terms of average delay $d = 1/(n\mu - l\mu) < D$, where n is the number of active servers, and μ represents the processing ability of servers. Zhang et al. made similar assumptions and imposed a QoS constraint in the form $t + 1/(s\mu - L) < t_D$, where t is an additional transmission delay [16].

Table 1. Comparison with related works on data center participation in demand response.

	Data center model with heterogeneous workload	Data-center job QoS optimization	Multi-data-center joint optimization	Aggregator social cost optimization
Liu, 2013 [7]			✓	
Lei, 2015 [8]	✓	✓		
Zhou, 2015 [30]		✓	✓	
Yu, 2017 [10]		✓	✓	
Cupelli, 2018 [12]		✓		
Zhou, 2018 [11]			✓	✓
Zhang, 2019 [13]	✓	✓		
Tsiligkaridis, 2019 [14]		✓		✓
Zhang, 2020 [16]			✓	✓
Niu, 2021 [18]			✓	
Zhang, 2021 [17]	✓	✓		
Jahanshahi, 2022 [19]	✓	✓		
Ours	✓	✓	✓	✓

Although using these lumped parameters facilitates the formulation and the solution of the corresponding optimization problems, the models we outlined ignore the heterogeneity of workloads and the complexity of the job scheduling process in data centers. In a data center with heterogeneous workload traces like ours, where different types of jobs have different execution times, number of servers required, job arrival rates, and QoS constraints, the heterogeneity of workloads cannot be easily characterized by a single workload parameter, and the complexity of the scheduling process of heterogeneous workloads also cannot be captured by a single data center utilization value.

In contrast, our work develops a data center model with workload heterogeneity, and we quantify job performance by both queueing-theoretic results and precisely modeled event-based job scheduling simulations where key parameters of our model (including server power and workload properties) are collected from a real data center.

Table 1 summarizes the comparison between our work and other related works. As discussed above, our work not only solves a joint optimization problem considering all three aspects, i.e., the data-center-side job QoS optimization, the multi-data-center workload sharing optimization, and the aggregator-side social cost optimization, but also models data centers with heterogeneous workloads which is closer to the job scheduling of data centers in practice.

3. Data-Center-Aggregator optimization framework (DCAopt)

DCAopt considers interactions among an aggregator and a set of S data centers. The aggregator intends to utilize the data centers' flexibility in power consumption via the broadcasting of a price incentive p_i to each data center i. This price represents a cost per unit of power used and is obtained via the minimization of a *social cost* metric, which quantifies the cost associated with not achieving the needed power reduction or the QoS target. This social cost metric has been used in the literature (such as Ref. [4]) to assess how well a desired load reduction could be achieved in a demand response scenario involving an aggregator and a set of active data centers. To minimize social cost, the aggregator needs to incentivize the data centers to reduce their power consumption by an amount T, which is decided upon *a priori* by an Independent System Operator (ISO). With this price incentive broadcast, the data centers collectively solve a global optimization problem to determine the number of active servers in each data center, as well as the arrangement of workload sharing. In addition, each data center also solves a local optimization problem to determine the optimal scheduling choice with constraints on QoS degradation. Figure 1 displays the interactions between the aggregator and a set of data centers (DCs). In the following, we first introduce our data center model, and then, we discuss three integrated optimization problems.

3.1 The data center model in DCAopt

In this work, we consider data centers that run workloads composed of several different types of jobs, labelled from job type 1 to job type K.

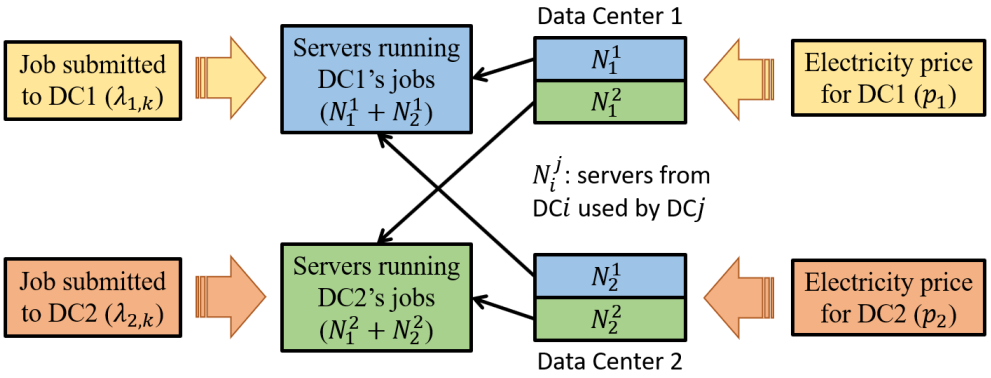

Figure 3. Workload sharing enables servers from one data center to run jobs that are submitted to another data center. With workload sharing, electricity costs and job QoS degradation can be reduced. DC is the abbreviation for data center.

Different job types have different characteristics such as job size, execution time, and arrival rate. Here, "job size" refers to the number of nodes required to run the job in parallel, and "arrival rate" refers to how many jobs of this type are submitted to the data center per unit time. We assume a data center uses separate queues to serve each type of job; so, for K job types, there are K different queues in each data center. The number of job types, K, can be different from one data center to another.

To serve these different types of jobs in a data center, we partition all servers into $K+1$ groups, including one idle group and K active groups. Each active group of servers processes all the jobs in one of the queues. In each active group, jobs are processed following the First-Come-First-Serve (FCFS) policy. The partition between active and idle groups is done conceptually instead of physically, and servers switch between the idle group or active groups from time to time depending on whether there are jobs to be processed. When there are no jobs, all servers are in the idle group; when there are jobs of a certain type to be processed, some idle servers switch into a corresponding active group. This model can reflect workload dynamics in many real-world data centers where multiple types of jobs exist and many instances of the same job type are frequently submitted by users. Figure 2 shows the model of a data center following the description above.

In order to balance the delay in all the queues in a controlled way, we apply the Generalized Processor Sharing (GPS) algorithm [31] to control how many servers are assigned to each active server group. The GPS algorithm was originally proposed to provide balanced performance in network scheduling. It provides a solution to situations where incoming requests from several separate queues are processed by a fixed amount of resource. The GPS algorithm assigns a non-negative weight w_k $(k = 1,2,\dots)$ to each queue, and the weights determine the portion of resource allocated to each queue. These weights are fixed at the beginning, and they satisfy $\sum_k w_k = 1$. When all the queues have waiting requests, w_k is exactly the portion of resource allocated to each queue. When some of the queues are empty (i.e., when they have no waiting jobs), their weight portions are redistributed to other non-empty queues following the assigned weights. In our data center scheduling problem, the GPS algorithm controls the number of servers in each active server group according to the weights.

3.2 Data center Quality-of-Service (QoS) optimization

In a data center, a key metric in quantifying the performance of a job is its queueing time in the system, T_{queue}, defined as the duration from the job's submission to its execution starting time. As each type of job has its own typical execution time, T_{exec}, a typical approach to quantify the performance of a job is to compare a job's queueing time with its execution time [32]. To that end, we define the *QoS degradation* of a job as $Q = T_{queue}/T_{exec}$. Since each queue in the data center contains many jobs of the same type, we are concerned with whether the majority of jobs of a type have their QoS degradation bounded. In other words, for each job type k, we want to ensure the ratio of jobs whose QoS degradation exceeds Q_{thres}^k to be less than δ^k. Here, Q_{thres}^k is a QoS degradation threshold for job type k, and δ^k is a ratio. In the following, we set $\delta^k = 10\%$, which means that we require 90% of each type of jobs' QoS degradation to be bounded by Q_{thres}^k.

To achieve the QoS constraints discussed above, we need to minimize the probability of QoS-degraded jobs, $\text{Prob}[Q^k > Q_{thres}^k] - \delta^k$, for each job type k. Thus, the data center QoS optimization minimizes the probability of QoS-degraded jobs for all queues as follows:

$$\min_{w_k} C(w_k) = \sum_k \text{SoftPlus}\left(\rho\left(\text{Prob}[Q^k > Q_{thres}^k] - \delta^k\right)\right), \quad (1)$$

where $\text{SoftPlus}(x)$ represents the softplus function defined as $\ln(1 + e^x)$, which is a smooth approximation of the ramp function $\max(0, x)$. With an appropriate selection of the scale parameter ρ, this softplus function is close to 0 only when the jobs of type k meet their QoS constraints. We solve this optimization problem using projected gradient descent considering the constraints on the weights, $\sum_k w_k = 1$ and $w_k \geq 0$. By estimating the probability $\text{Prob}[Q^k > Q_{thres}^k]$, we can obtain an optimal selection of the weights.

To understand how $\text{Prob}[Q^k > Q_{thres}^k]$ depends on various parameters including weights, number of servers, and job arriving rates, we apply a queueing-theoretic result proven by Paschalidis et al. [33] [34], which states that the probability of large delays in a queueing system following the GPS algorithm should decrease exponentially as the delay Q_{thres}^k increases:

$$\text{Prob}[Q^k \geq Q_{thres}^k] = r_k e^{-Q_{thres}^k \cdot \theta_k^*}, \qquad (k = 1,2,\dots,K), \quad (2)$$

and most importantly, the coefficients θ_k^* can be theoretically calculated from the distributions of job arrival times and job processing times by:

$$\theta_k^* = \sup_{\theta \geq 0, \Lambda_{GPS,k}(\theta) < 0} -\Lambda_B(-\theta w_k), \quad (3)$$

$$\Lambda_{GPS,k}(\theta) = \Lambda_{A_k}(\theta) + \Lambda_B(-\theta w_k). \quad (4)$$

Here, the functions $\Lambda_{A_k}(\theta)$ and $\Lambda_B(\theta)$ are the logarithm of the moment-generating functions for random variables $A_k(t)$ and $B(t)$, where $A_k(t)$ is defined as the amount of workload (i.e., number of jobs

Table 2. Definition of notations used in this work.

Type	Label	Description
DCAopt inputs	α_i	Power usage per server in data center i
	β_i	Coefficient in QoS cost defined in Eq. (5)
	γ_i	Exponential in QoS cost defined in Eq. (5)
	κ	Service sharing cost per server in Eq. (5)
	η	Coefficient in penalty function in Eq. (8)
	S	Number of data centers
	N_i^U	Total number of servers in data center i
	T	Load reduction target
	$\lambda_{i,k}$	Job arrival rate of workload type k in data center i
	T_k	Execution time of workload type k
	m_k	Job size (# of servers) of workload type k
	Q_{thres}^k	QoS degradation threshold of job type k
	δ^k	Bound of QoS violation ratio in Eq. (2)
	ρ	Coefficient in QoS cost in Eq. (1)
DCAopt outputs	p_i	Electricity price for data center i
	N_i^j	The number of active servers in data center i that are used by jobs in data center j
	G	Social cost defined in Eq. (7)
Inter-mediate variables	$w_{k,i}$	The optimal weight for job type k in data center i when using the GPS algorithm (Index i is omitted in Section 3.2)
	r_k	Fitted parameter for job type k in Eq. (2)
	Q^k	QoS degradation of job type k
	θ_i	The exponential to quantify job QoS degradation in data center i in Eq. (5)
Algorithm hyper-parameters	a	Gradient descent step size in Algorithm 1
	b	Gradient descent step size in Algorithm 2
	Δ	Electricity price perturbation
	t_{max}	Max # of iterations in Algorithm 1 line 6
	u_{max}	Max # of iterations in Algorithm 2 line 1
	v_{max}	Max # of iterations in Algorithm 1 line 7

× job execution time × job size) arriving in the k^{th} queue per unit time at t, and $B(t)$ is defined as the total processing capability of the system at time t, i.e., the number of active servers for processing the jobs. The time variable t is discrete.

When applied to our data center model, $B(t)$ has a fixed value that is proportional to the number of active servers in the data center. The specific form of $\Lambda_{A_k}(\theta)$ depends on the distribution of job arrivals, which can be empirically estimated from historical workload traces. In the following, we assume the job arrival times in the k^{th} queue ($k \in \{1,2,\dots\}$) follow a Poisson distribution with parameter λ_k, and our experiments reported later are based on this assumption. Although we make the Poisson distribution assumption for evaluation purposes, our optimization framework can be applied to other job arrival distributions.

Following the Poisson distribution assumption and applying Eq. (3) and Eq. (4) to our data center model, for each job type k, we obtain $\theta_k^* = N\Theta_k w_k$, where N is the number of servers used by a data center, and Θ_k is a variable satisfying the equation $\lambda_k e^{\Theta_k m_k T_k} = N\Theta_k w_k + \lambda_k$. For each job type k, T_k is its typical execution time, and m_k is the job size. While θ_k^* can be estimated from queueing theory as shown above, it is hard to theoretically estimate the r_k parameter in Eq. (2), so we run data center simulations using our job scheduling simulator to empirically obtain its value.

This queueing-theoretic derivation of $\text{Prob}[Q^k > Q_{thres}^k]$ in Eq. (2) assumes that jobs are processed following the FCFS policy. In our data center model, each active server group processes jobs following the FCFS policy, so the derivation above can be adopted. If different scheduling policies are used, adjustments to the queueing-theoretic derivation may need to be considered.

3.3 Multi-data-center coordination in DCAopt

While each data center solves its own QoS optimization problem, all data centers also collectively solve a cost minimization problem shown in Eqs. (5) and (6) as follows:

$$\min_{N_i^j} \sum_{i=1}^{S} p_i \alpha_i \sum_{j=1}^{S} N_i^j + \kappa \sum_{i=1}^{S} \sum_{j \in \{1,\dots,S\} \setminus i} N_i^j + \sum_{i=1}^{S} \beta_i e^{-\gamma_i \theta_i(N_i)} \qquad (5)$$

$$\text{s.t.} \quad 0 \le \sum_{j=1}^{S} N_i^j \le N_i^U, \forall i. \qquad (6)$$

In Eq. (5), the cost function represents the sum of electricity costs, the workload-sharing costs, and job QoS-constraint violation costs from all data centers considered. The responses, $N_i^j, \forall i,j \in \{1,\dots,S\}$, represent the number of active servers in data center i that are used by jobs in data center j. When the superscript matches the subscript, this N_i^i represents servers in data center i that are not shared. Because the power usage of servers in each data center, α_i, can differ from each other, sharing workloads from one data center to another could be beneficial to the overall cost in Eq. (5). Workload sharing is also illustrated in Figure 3 where two data centers are included. Here, servers from one data center can be shared to run jobs that are submitted to another data center. For example, all jobs submitted to data center 1 run on $N_1^1 + N_2^1$ servers. Meanwhile, the electricity cost for data center 1 is charged on its $N_1^1 + N_1^2$ servers.

In the objective function of Eq. (5), the first term represents the electricity costs of the servers in all data centers. In this term, p_i is the electricity price for data center i, so this term favors a smaller number of active servers.

The second term represents workload-sharing cost, with coefficient κ quantifying the per-server sharing cost, which includes extra efforts needed in cross-platform workload management, as well as extra resource usage due to workload transfer. Note that this assumption neglects the difference in the amount of data being transferred, so it can be viewed as a first-order approximation to the actual cost. Especially, for the compute-intensive workloads which include most physics simulation workloads such as molecular dynamics and fluid dynamics, this assumption should work well as these workloads rely on solving equations numerically and are usually not data-intensive. Similar assumptions are also made by previous works such as Ref. [11].

The final term represents the cost for job QoS-constraint violation in each data center. Here, we assume this cost takes an exponential form following the large delay probability in Eq. (2). Variable $\theta_i(N_i)$ is dependent on N_i, and the latter is defined as the total number of servers that can be used by data center i's jobs, hence $N_i = \sum_{j=1}^{S} N_j^i$. Variable $\theta_i(N_i)$ is estimated following the same method in Eq. (3), and β_i and γ_i are two coefficients. The constraints in Eq. (6) ensure that the total number of active servers in data center i is bounded by its server capacity, N_i^U.

Solving the optimization problem in Eq. (5) returns the optimal arrangement of active servers that minimizes the cost. In our numerical evaluation, we apply projected gradient descent to find the optimal solution, where θ_i is dependent on N_i and is computed as the average of θ_k^* over all job types k in data center i. Given N_i, we can calculate θ_i by solving the QoS optimization problem in Eq. (1).

3.4 Aggregator optimization

While data centers pursue the target of reducing their cost and QoS degradation, the aggregator aims at achieving a load decrease amount T to handle a supply deficit or demand surplus event. To incentivize data centers to meet the load decrease target, the aggregator selects the optimal electricity prices that incentivize demand response, obtained via the minimization of a social cost metric $G(\boldsymbol{p})$ shown in Eq. (7), and the price p_i is broadcasted to data center i:

$$\min_{\boldsymbol{p}\geq\boldsymbol{0}} G(\boldsymbol{p}) = q\left(T - \sum_{i=1}^{S}\alpha_i\left(N_i^U - \sum_{j=1}^{S} N_i^j(\boldsymbol{p})\right)\right) + \sum_{i=1}^{S}\beta_i e^{-\gamma_i\theta_i(N_i(\boldsymbol{p}))}, \qquad (7)$$

where

$$q(x) = \frac{\eta x^2}{2}. \qquad (8)$$

The first term in $G(\boldsymbol{p})$ is a penalty for not achieving the desired target load reduction T, where $q(x)$ is defined to be a convex penalty function. For our experiments, we assume q is a quadratic function defined in Eq. (8). The second term in $G(\boldsymbol{p})$ represents a social welfare cost quantified by the QoS violations in the data centers. The dependence of $N_i^j(\boldsymbol{p})$ and $N_i(\boldsymbol{p}) = \sum_{j=1}^{S} N_j^i(\boldsymbol{p})$ on the price vector $\boldsymbol{p}$ is emphasized in Eq. (7), and the variables $N_i^j(\boldsymbol{p})$ are calculated by solving the multi-data-center coordination problem in Eq. (5).

Algorithm 1: Data-Center-Aggregator Optimization

Input: t_{max}, v_{max}, $\boldsymbol{\Delta}$, a, and the parameters in Eqs. (1)-(8)
Output: optimized broadcast prices $\boldsymbol{p}$, optimized server arrangement $N_i^j\ \forall i,j$

1 **For** i from 1 to S **do**
2 Perform event-based job scheduling simulation of data center i with the current values of variables
3 Use QoS-degradation probability $\text{Prob}[Q^k > Q_{thres}^k]$ from simulation to fit r_k and calculate θ_k^* in Eq. (2)
4 Apply Algorithm 2 to solve the QoS optimization problem in Eq. (1) for data center i and obtain $w_{k,i}$
5 **endfor**
6 **For** t from 1 to t_{max} **do**
7 Apply v_{max} iterations of PGD to solve the multi-data-center coordination problem in Eq. (5)
8 Calculate $G(\boldsymbol{p})$ in Eq. (7) using optimized N_i^j from line 7
9 **For** each price component i from 1 to S **do**
10 Perturb price component by $\boldsymbol{p}' \leftarrow \boldsymbol{p} + \boldsymbol{\Delta}$ and apply PGD to solve Eq. (5) again using $\boldsymbol{p}'$
11 **endfor**
12 Calculate $G(\boldsymbol{p}')$ and approximate $\nabla G(\boldsymbol{p}) = \frac{\partial G}{\partial \boldsymbol{p}}$ via finite difference between $G(\boldsymbol{p}')$ and $G(\boldsymbol{p})$
13 Conduct one iteration of PGD for the aggregator optimization in Eq. (7) by $\boldsymbol{p} \leftarrow [\boldsymbol{p} - \boldsymbol{a}\cdot\nabla G(\boldsymbol{p})]^+$
14 **endfor**
15 **Return** the optimized broadcast prices $\boldsymbol{p}$, and the optimized server arrangement N_i^j

In Eq. (7), we assume that data centers truthfully interact with the aggregator and provide accurate server response values obtained from Eq. (5). The objective of Eq. (7) contains a social welfare term that is the aggregate QoS cost of all data centers. Thus, the prices that the aggregator solves for are obtained with the data center costs in mind; if the data centers seek to manipulate their responses to attempt to gain larger price incentives, this would be difficult to achieve as the data centers are not aware of the aggregator penalty term in Eq. (7) that will directly be affected by the altered data center responses. This will result in the aggregator not being able to actually achieve its target reduction and the data centers receiving suboptimal incentives. With this in mind, there is incentive for data centers to provide accurate information. As future work, regulatory and/or compliance constraints can directly be imposed to ensure veracity of data centers responses. A game-theoretic framework, that by design can encourage truthful data center interactions [20], can also be explored to model the data center-aggregator dynamic.

For readers' convenience, we list the variables we use in Table 2.

3.5 Algorithms in DCAopt for solving the optimization problems

We use Projected Gradient Descent (PGD) as our solution method in the following algorithms for solving the optimization problems. We choose PGD as it is a commonly used and easily implementable approach to solving constrained optimization problems. PGD is an extension of the conventional Gradient Descent approach to solving unconstrained optimization problems; with this approach, there is an additional projection step that must be done to place the obtained estimate in the feasible set of solutions characterized by the constraints of the optimization problem. For PGD to converge, the objective of the optimization problem at hand must be convex, the constraint set of the optimization problem must be convex, and an appropriate step size must be used for the gradient updates [35]. With these conditions, convergence can be attained, and no numerical instability will arise. The PGD conditions are met in our optimization tasks.

Algorithm 1 shows how the aggregator and the set of S data centers interact with each other via an iterative process, where the goal of the interaction is to obtain the optimized broadcast prices $\boldsymbol{p}$ (i.e., p_i for i from 1 to S) and the data center server arrangement N_i^j, for all i,j.

The algorithm starts with some initialized $\boldsymbol{p}$ and N_i^j values, chosen within the feasible region. In lines 1-3, we simulate the workload processing of each data center to obtain the QoS-degradation probability $\text{Prob}\left[Q^k > Q_{thres}^k\right]$, and by comparing it with the theoretical value derived from Eq. (2), we fit the parameter r_k for each data center. Then, in line 4, we solve the data center QoS optimization problem in Eq. (1) using PGD for each one of the data centers, which is detailed in Algorithm 2. In lines 7-8, we solve the multi-data-center coordination problem in Eq. (5) to get $N_i^j(\boldsymbol{p})$. Because minimizing Eq. (5) follows a similar PGD optimization process as Algorithm 2, we omit the detailed pseudocode. Next, in lines 9-13, we perturb the prices by a small amount $\boldsymbol{\Delta}$ to get $\boldsymbol{p}' \leftarrow \boldsymbol{p} + \boldsymbol{\Delta}$ and solve Eq. (5) again for each price component. By comparing the social cost in Eq. (7) calculated before and after the price perturbation, i.e., $G(\boldsymbol{p})$ and $G(\boldsymbol{p}')$, we can estimate the gradient $\frac{\partial G}{\partial \boldsymbol{p}}$ using a finite difference approximation, which is then used in performing the PGD to solve the aggregator optimization problem in Eq. (7). At each iteration of PGD, we update prices by $\boldsymbol{p} \leftarrow [\boldsymbol{p} - \boldsymbol{a}\cdot\nabla G(\boldsymbol{p})]^+$. When the PGD stabilizes, the prices $\boldsymbol{p}$ we obtain are the optimized broadcasted prices, and the corresponding N_i^j correspond to the optimized server arrangement.

In Algorithm 1, we use parameter t_{max} to denote the maximal number of iterations in performing PGD for the aggregator optimization and use a to denote the step size in PGD. In line 13, $[\boldsymbol{p} - \boldsymbol{a}\cdot\nabla G(\boldsymbol{p})]^+$

means projecting ($\boldsymbol{p} - \boldsymbol{a} \cdot \nabla G(\boldsymbol{p})$) onto the feasible region where the values in every dimension are non-negative.

Algorithm 2 details the data center QoS optimization part in line 4 of Algorithm 1. PGD is applied here to minimize $C(w_{k,i})$ in Eq. (1) through a step-by-step adjustment of weight parameters $w_{k,i}$. In Algorithm 2, b is the step size of gradient descent. Inside the cost function $C(w_{k,i})$, the scale parameter ρ controls the smoothness of the SoftPlus function. If ρ is too small, the QoS degradation probability $\text{Prob}[Q^k > Q^k_{thres}]$ in Eq. (1) is more likely to surpass the constraint δ^k. On the other hand, if ρ is too large, the derivative $\frac{\partial C}{\partial w_{k,i}}$ is steeper, and it will take more time for the gradient descent optimization to reach the optimum. For our experiments, we considered different values and empirically determined $\rho = 100$ to be a good choice in terms of achieving both the validity of constraints and the feasibility of optimization.

Algorithm 2: Data Center QoS Optimization

Input: r_k, b, u_{max}, and other parameters in Eqs. (1)-(8)
Output: optimal weights $w_{k,i}$ for job type k in data center i
1 **For** u from 1 to u_{max} **do**
2 Calculate the cost function $C(w_{k,i})$ in Eq. (1) using Eqs. (2)-(4) and the current values of variables
3 Calculate derivatives $\nabla C(w_{k,i}) = \frac{\partial C}{\partial w_{k,i}}$
4 Apply gradient descent to update the weight parameters $w_{k,i}$ by $w_{k,i} \leftarrow w_{k,i} - b \cdot \nabla C(w_{k,i})$
5 Project the updated weights onto the $\sum_k w_{k,i} = 1$ plane within the region $w_{k,i} \geq 0$
6 **endfor**
7 **Return** the optimal weight parameters $w_{k,i}$

Computational complexity: Based on the algorithms discussed above, we analyze the computational complexity of our DCAopt framework. Executing lines 2-3 of Algorithm 1 runs data center job scheduling simulations, which takes $T_{simu} \approx 10s$ when simulating the job scheduling of the one-hour period of a data center with hundreds of nodes. Thus, the complexity of lines 2-3 is $S \cdot T_{simu}$. Line 4 of Algorithm 1 corresponds to Algorithm 2. The fact that we have an analytical form of $C(w_{k,i})$ based on queueing theory allows us to calculate $C(w_{k,i})$ and its derivatives in constant time. Thus, the complexity of Algorithm 2 for each data center is proportional to the number of PGD iterations, which is $O(u_{max})$. The cost function in Eq. (5) is also analytical, so the complexity of lines 7-8 in Algorithm 1 is also proportional to the number of PGD iterations at this step, which is $O(v_{max})$. Next, in lines 9-11 of Algorithm 1, $O(S \cdot v_{max})$ time is spent on doing the multi-data-center optimization for each component of the perturbed price $\boldsymbol{p}'$. Finally, lines 6-14 is executed for t_{max} times for applying PGD on the aggregator optimization.

Therefore, from the above analysis, we see that the time complexity of Algorithm 1 is $O\big(S \cdot (T_{simu} + u_{max} + v_{max} \cdot t_{max})\big)$, which scales linearly with the number of data centers S. For our experiments to be discussed in the following section, the PGD optimization always converges in 30 iterations, so we can set $u_{max} = v_{max} = t_{max} = 30$, and the entire execution of DCAopt for our setting takes less than a few minutes. Since the aggregator optimization depends on the power reduction target value, Algorithm 1 should be executed every time when the aggregator broadcasts a new power reduction target to data centers, which could be either per-hour, per-day, or per-week, based on the specific contract between the aggregator and the data centers. Therefore, as the algorithm is only executed once at the times the power reduction target gets updated, the temporal cost of applying our algorithm is acceptable.

4. Experimental methodology

To evaluate the performance of our DCAopt framework, we create some test cases where one aggregator interacts with two data centers. We implement Algorithms 1 and 2 in MATLAB to solve all optimization problems, and we build an event-based data center simulator to conduct simulation of the job scheduling and execution taking place in the data centers. The data center job scheduling simulation gives the job QoS degradation values for us to fit the r_k variables in Eq. (2), which are used in Algorithm 2 to conduct the data center QoS optimization. The latter results are further used in Algorithm 1 to conduct the multi-data-center coordination and the aggregator optimization. Note that the ability of our framework to optimize data center QoS is achieved by applying the approach in Section 3.2, which does not rely on the implementation details of the job scheduling simulator. In fact, this scheduling simulator can be replaced by any simulator that can take a heterogeneous workload trace and simulate multiple queues.

Scheduling simulator: We build our event-based data center job scheduling simulator in Python. This simulator works as follows: given a workload trace together with the job properties (job size and job execution time), when the simulated time goes second-by-second, the simulator records or changes the activity state (active/idle) of servers based on the starting/ending event of a job. The simulator maintains a queue for each type of submitted jobs waiting to be executed. Every second, the simulator updates the number of servers running each type of jobs and checks the number of waiting jobs in each queue. A starting event of a job happens when the job is able to run following the GPS scheduling algorithm (Section 3.1). In other words, when there are waiting jobs in a queue and the ratio of active servers running this type of jobs is below the corresponding weight w_k, the simulator allows a job of this type to start execution as long as there are enough idle servers. When a job's preset execution time is reached, the simulator records the ending event of the job. Finally, after a simulation run of a workload trace finishes, we calculate the total electricity cost of this run and calculate the average QoS degradation of jobs based on their recorded starting and ending times.

Table 3. Properties of all types of jobs used in our experiments.

Job type ($\boldsymbol{k}$)	Job type name	Execution time ($\boldsymbol{T_k}$)	Job size ($\boldsymbol{m_k}$)	Job arrival rate in DC1 ($\boldsymbol{\lambda_{1,k}}$)	Job arrival rate in DC2 ($\boldsymbol{\lambda_{2,k}}$)
1	MG.D.16	84 s	1	0	0.238
2	SP.C.16	54 s	1	0	0.371
3	IS.D.32	42 s	3	0.079	0.159
4	BT.D.49	551 s	2	0.009	0
5	EP.D.64	54 s	3	0	0.123
6	CG.C.4	28 s	1	0	0.719
7	MG.D.8	141 s	1	0.071	0
8	IS.D.4	122 s	1	0.082	0
9	LU.C.28	29 s	1	0	0.690
10	EP.D.100	36 s	4	0.069	0
11	IS.D.64	27 s	4	0.092	0.183
12	LU.D.112	164 s	4	0.015	0
13	MG.D.32	49 s	2	0.102	0.204

Data center properties: Our simulation adopts data center properties and workload properties (including job size, job execution time, server

Table 4. Understanding the impact of input parameters by comparing seven different scenarios.

Scena-rios	Inputs			Outputs								
	κ (\$/h)	β_2 (\$/h)	η (\$/W^{2}h)	N_1^1	N_1^2	N_2^1	N_2^2	p_1 (\$/kWh)	p_2 (\$/kWh)	Power reduction (kW)	Social cost before optimization	Social cost after optimization
1	0.030	5	10^{-7}	115	35	0	119	0.062	0.198	60.5	25.1	9.3
2	0.032 (↑)	5	10^{-7}	121	12	0	144	0.096	0.181	57.6	25.9	12.1
3	0.000 (↓)	5	10^{-7}	96	54	2	107	0.061	0.158	63.8	8.3	7.2
4	0.030	7 (↑)	10^{-7}	110	40	0	122	0.062	0.212	59.5	32.7	10.8
5	0.030	2.5 (↓)	10^{-7}	130	20	0	114	0.060	0.166	62.1	11.7	7.0
6	0.030	5	10^{-6} (↑)	102	48	0	97	0.095	0.238	67.8	222.9	8.7
7	0.030	5	5×10^{-8} (↓)	128	22	0	137	0.060	0.169	54.4	14.1	9.9

power usage, etc.) that are measured by us at an actual operational data center, the Massachusetts Green High Performance Computing Center (MGHPCC), which is a large data center providing high-performance computing resources for multiple universities in Massachusetts (Boston University, Harvard, MIT, Northeastern, and Univ. of Massachusetts). Each server that we experiment with in MGHPCC contains two Intel Xeon Gold 6132 CPUs, and each CPU has 14 cores. Each CPU's thermal design power is 140 W. In our test cases, we assume there are two data centers each having $N_1^U = 150$ and $N_2^U = 300$ servers. We assume the active power consumptions per server are as follows: $\alpha_1 = 322$ Watts for data center 1 and $\alpha_2 = 334$ Watts for data center 2, which are averaged values measured from servers in MGHPCC using the IPMI tool [36] when running two different sets of workloads discussed below.

Experiment parameters: We assume the ISO sets the load reduction target as $T = 70$ kW, which is an achievable amount if half of the servers in both data centers are turned off. We assume the following default price parameters are used in our experiments: $\kappa = 0.03$ \$/h, $\beta_1 = \beta_2 = 5$ \$/h, and scenarios with altered price parameters are explored in Section 5.2. The default coefficients in the cost functions in Eqs. (5) and (7) are set as $\gamma_1 = \gamma_2 = 10$ and $\eta = 10^{-7}$, and altering the η parameter is explored in Scenario 6. At the beginning of executing Algorithm 1, we initialize the electricity prices for the two data centers as $p_1 = 0.060$ \$/kWh and $p_2 = 0.150$ \$/kWh. The price perturbation is set as $\Delta = 0.02$ \$/kWh for each price component.

Workload properties: Our data center scheduling simulation runs 13 different types of jobs, and all the jobs are selected from the benchmark applications in the NAS Parallel Benchmark Suite [37]. The properties of the jobs and their arrival rates used in our experiments are listed in Table 3. In the Job Type Name column, MG, SP, IS, etc., refers to benchmark applications in the NAS Parallel Benchmark. The suffix in the name refers to the input and the number of threads used in running the job. For example, "MG.D.16" represents the MG benchmark application with input D and running with 16 threads. The execution times listed in the table are real values measured when we run the jobs on a certain number of nodes (i.e., job size m_k) at the MGHPCC. The job arrival rates in Table 3, $\lambda_{1,k}$ and $\lambda_{2,k}$ (number of jobs arriving per second), are determined by assuming the data centers' server utilization rates are approximately 50%. In the table, some job types have a zero arrival rate, which means that these job types appear in only one of the two data centers.

Given these workload properties, a workload trace is generated for each data center assuming the jobs' arrival times follow a Poisson distribution, where the job arrival rates for each type of jobs in Table 3 are determined by assuming the data centers' server utilization rates are approximately 50%.

5. Results

In this section, we present our evaluation results for DCAopt. We first provide detailed analysis on a default test case in Section 5.1. Then, we compare the default case with six other scenarios with altered parameters in Section 5.2.

5.1 Best price setting and server arrangement in the test case

Applying our DCAopt framework on the test case (referred to as Scenario 1) with parameters discussed in the previous section, we obtain the best electricity price setting as $p_1 = 0.062$ \$/kWh, $p_2 = 0.198$ \$/kWh. The corresponding best server arrangement is $N_1^1 = 115, N_1^2 = 35, N_2^1 = 0, N_2^2 = 119$, which means keeping all the 150 servers in data center 1 active, among which 115 servers are employed to run data center 1's jobs, and 35 servers are shared for data center 2 to run that data center's jobs. Meanwhile, only 119 servers of data center 2 are kept active to run jobs, and no server from data center 2 is used to run data center 1's job, which also reflects the electricity price difference between the two data centers. Since the electricity price for data center 1 is much lower than that for data center 2, prioritizing the use of data center 1's servers reduces the electricity cost.

On the aggregator side, the best electricity prices determined by DCAopt reduces the social cost in Eq. (7). If applying the initialized prices, $p_1 = 0.060$ \$/kWh and $p_2 = 0.150$ \$/kWh, the power reduction we can achieve is only 49 kW, and the corresponding social cost is 25.1. Instead, applying the best prices enables us to achieve a power reduction of 60.5 kW, corresponding to a 23% increase in power reduction, and the social cost using the best prices decreases to 9.3. Figure 4 shows how the social cost value and the best server

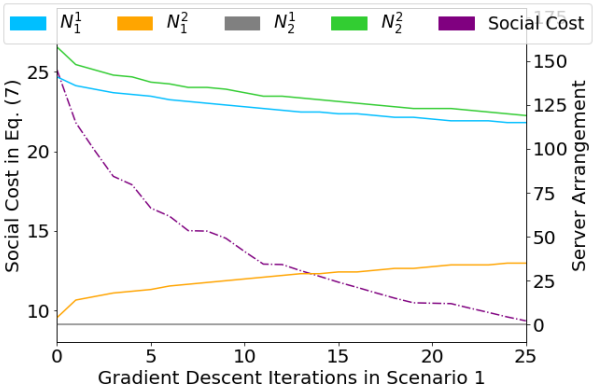

Figure 4. Social cost decreases and server arrangements (N_i^j) approach their best values as we apply our algorithms and optimize the cost function in Eq. (7) through iterations of gradient descent.

arrangement change through iterations of gradient descent when we optimize the cost function in Eq. (7). From Fig. 4, we see our algorithms quickly converge in tens of iterations.

5.2 Comparing the best settings in different scenarios

To understand how the best setting found by DCAopt changes with the input parameters, in the following, we compare the default case with other scenarios where each scenario alters some input parameters. Table 4 summarizes the input parameters we use in these scenarios as well as the results returned by DCAopt, which are discussed in the following.

Parameter κ: Since κ controls the per-server sharing cost in Eq. (5), we expect that a larger κ curbs workload sharing. In fact, if we increase κ from 0.03 \$/h to 0.032 \$/h (referred to as Scenario 2) and apply our DCAopt framework, the number of shared servers, N_1^2, decreases from 35 to 12. Further increasing κ to 0.033 \$/h makes the optimal number of shared servers to be zero, which demonstrates that a workload sharing cost that is too high could curb any attempts in workload sharing between data centers. On the other hand, reducing workload sharing cost promotes the sharing. For example, when we decrease κ to 0.025 \$/h, the optimal number of shared servers is $N_1^2 = 38$. Further decreasing κ to zero (referred to as Scenario 3) also motivates data center 2 to share servers to run data center 1's jobs, and in that case, the optimal server arrangement found by DCAopt is $N_1^1 = 96, N_1^2 = 54, N_2^1 = 2, N_2^2 = 107$.

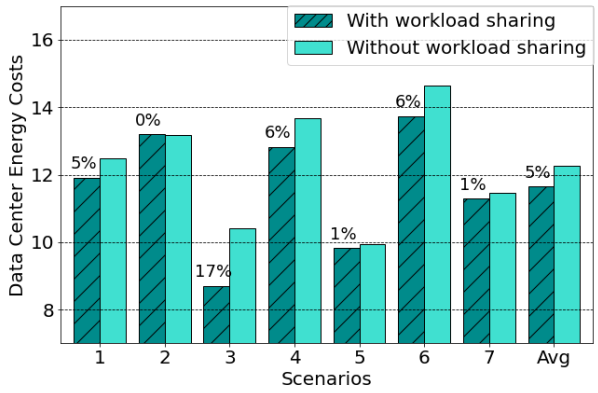

Figure 5. Comparing data center energy costs with and without workload sharing. The energy costs of data centers (including workload sharing costs) can be reduced by 5% on average when applying DCAopt.

Parameter β_i: Since β_1 and β_2 reflect the cost regarding the QoS degradation of jobs in data center 1 and data center 2, a larger penalty on QoS degradation motivates data centers to allocate more servers to run the submitted jobs. For example, if we increase β_2 from 5 \$/h to 7 \$/h (referred to as Scenario 4) in our test case while having all other parameters remaining the same (including $\kappa = 0.03$ \$/h), then the optimal server arrangement found by DCAopt is $N_1^1 = 110, N_1^2 = 40, N_2^1 = 0, N_2^2 = 122$. As expected, the number of servers employed to run data center 2's jobs increases to $N_1^2 + N_2^2 = 162$ to match the increased penalty on data center 2's job's QoS degradation. On the other hand, if we decrease β_2 to 2.5 \$/h (referred to as Scenario 5), the optimal server arrangement is $N_1^1 = 130, N_1^2 = 20, N_2^1 = 0, N_2^2 = 114$, and the number of servers employed to run data center 2's jobs decreases to $N_1^2 + N_2^2 = 134$. As a result, the θ_2 variable that quantifies the QoS degradation of data center 2's jobs decreases from $\theta_2 = 0.076$ in Scenario 1 to $\theta_2 = 0.015$ in this case (smaller values represent larger QoS degradation). Meanwhile, data center 1's jobs' QoS improves as θ_1 increases from 0.072 in Scenario 1 to 0.104 because less servers from data center 1 need to be shared. Further decreasing β_2 could lead to a point where QoS is ignored and the number of active servers is kept so low that the job waiting time in the queues diverges, which is the case when $\beta_2 < 2$ \$/h.

Parameter η: In the social cost G defined in Eq. (7), η reflects how we balance the two objectives, meeting the power reduction target, or meeting the QoS requirement. Increasing η will prioritize meeting the power reduction target. For example, if we increase η from 10^{-7} to 10^{-6} (referred to as Scenario 6), the power reduction we achieve increases from 60.5 kW to 67.8 kW, which becomes much closer to the preset power reduction target $T = 70$ kW. In the meantime, as a larger power reduction is prioritized, the QoS of jobs in the data centers is sacrificed. With $\eta = 10^{-6}$, the optimal server arrangement found by DCAopt is $N_1^1 = 102, N_1^2 = 48, N_2^1 = 0, N_2^2 = 97$, where the number of servers used to run either of the two data centers' jobs decreased compared to Scenario 1. The θ_i variables that quantify QoS degradation also decrease from $\theta_1 = 0.072, \theta_2 = 0.076$ (Scenario 1) to $\theta_1 = 0.045, \theta_2 = 0.047$ (Scenario 6), which represents a decrease in data center QoS as the cost of reaching a larger amount of power reduction. On the other hand, when η is decreased to 5×10^{-8} (referred to as Scenario 7), the achieved power reduction decreases to

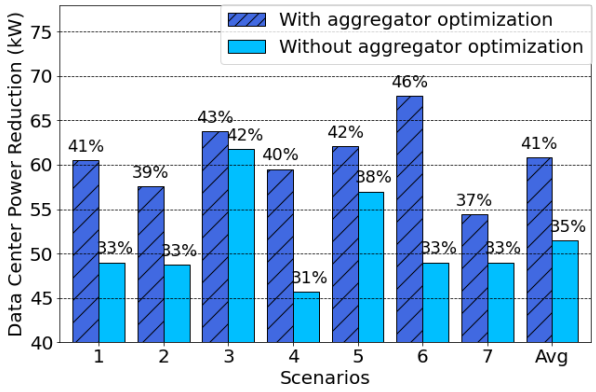

Figure 6. Improvement on data center power reduction when applying DCAopt. Data centers achieve a power reduction of 37%-46% when the aggregator applies the optimal electricity prices determined by DCAopt. These reductions are 6% higher than conducting only data-center-side optimization.

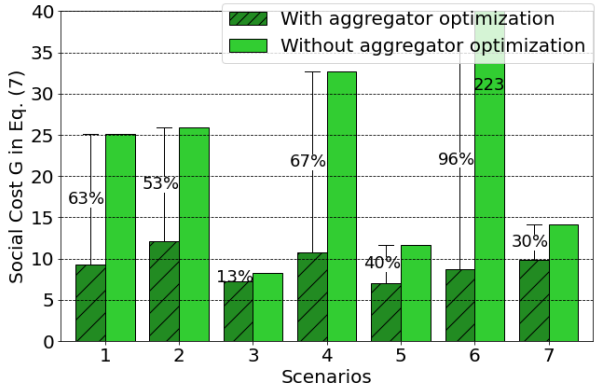

Figure 7. Comparing the social cost with or without the aggregator optimization in different scenarios. Applying the electricity prices determined by DCAopt reduces the overall social cost by more than 30% in most cases.

54.4 kW. In the meantime, the θ_i variables increase to $\theta_1 = 0.099$, $\theta_2 = 0.093$, representing an improvement on data center QoS.

Figures 5-7 summarize the seven scenarios discussed above. Figure 5 compares the data center energy costs with and without workload sharing. In the case of two data centers, the energy costs with workload sharing are calculated by

$$Cost_{share} = p_1\alpha_1(N_1^1 + N_1^2) + p_2\alpha_2(N_2^1 + N_2^2) + \kappa(N_1^2 + N_2^1),$$

where the term $\kappa(N_1^2 + N_2^1)$ represents sharing cost. On the other hand, without workload sharing, the energy costs of the data centers to meet the same QoS become

$$Cost_{noshare} = p_1\alpha_1(N_1^1 + N_2^1) + p_2\alpha_2(N_1^2 + N_2^2).$$

Note that in this case the electricity price for the N_2^1 servers is now p_1 instead of p_2. By comparing the two energy costs calculated above, in Fig. 5, we see the energy costs of data centers can be reduced by 5% on average when applying workload sharing using DCAopt.

Figure 6 quantifies the improvement on data center power reduction when applying DCAopt. Each bar in Fig. 6 represents the amount of power reduction when applying DCAopt relative to the case when keeping all servers active in the two data centers. In all scenarios, applying the aggregator optimization enables us to achieve larger power reduction while considering the QoS of jobs. Figure 6 shows that DCAopt reduces power consumption of data centers by 37-46% in our experiments, achieving a significant boost in power reduction compared to the 31-42% reduction achieved by conducting only data-center-side optimization. On average, the power reduction can be boosted by 6% when applying DCAopt.

Figure 7 quantifies the reduction on the overall social cost defined in Eq. (7) when applying DCAopt. In Fig. 7, the bars compare the social costs when applying either the initial electricity prices or the optimized electricity prices obtained by DCAopt. In most scenarios, applying the optimized prices reduces the social cost by more than 30%. There is a large reduction in social cost in Scenario 6, which is because the social cost in this scenario is much larger than the other scenarios due to the large $\eta = 10^{-6}$ (compared to $\eta = 10^{-7}$ in Scenarios 1-5), and this large η coefficient motivates data centers to reduce more power.

6. Conclusion and future work

In this work, we proposed a Data-Center-Aggregator optimization framework, DCAopt, that provides an optimized strategy for data centers to participate in demand response and for aggregators to achieve desired power reductions. By evaluating our framework on multiple test cases through data center simulations using heterogeneous workload traces, we demonstrate that our framework enables the aggregator to approach its power reduction target by assigning the best electricity prices to the data centers. On the data center side, DCAopt enables the data centers to participate in demand response programs to minimize their electricity costs while having the Quality-of-Service of jobs under constraints. With DCAopt, the energy costs of data centers can be reduced by 5% on average, with a corresponding reduction of the social cost amounting to more than 30% in most cases. In addition, power usage reduction at the data centers is 6% higher compared to data-center-centric power use optimization.

In future work, a more expansive formulation involving generators, flexible loads (such as data centers), and inflexible loads (rigid entities that must use a known amount of power) can be considered. We can also consider a model where workload shifting over time is permitted to yield additional reduction benefits. Additionally, this work does not consider the influence of data centers' service prices on job submission rates. Incorporating this interaction between data centers and their users requires data centers to optimize their service prices considering the response of job arrival rates to the prices.

Conflict of interest

The authors declare that there is no conflict of interest.

Declaration of competing interest

The authors report no declarations of interest.

Acknowledgements

This research is partially supported by the NSF under grants CCF-2200052, IIS-1914792, and DMS-1664644, by the ONR under grant N00014-19-1-2571, and by the Boston University College of Engineering under the Dean's Catalyst Award.

Author contributions

Conceptualization: ICP and AKC. Methodology: YZ, AT, ICP, and AKC. Coding: AT and YZ. Experiments: YZ. Visualization: YZ. Writing: YZ, AT, ICP and AKC.

References

[1] Shehabi, A., Smith, S., Sartor, D., Brown, R., Herrlin, M., Koomey, J., ... & Lintner, W. (2016). United states data center energy usage report. Technical Report No. LBNL-1005775, Lawrence Berkeley National Laboratory.

[2] Masanet, E., Shehabi, A., Lei, N., Smith, S., & Koomey, J. (2020). Recalibrating global data center energy-use estimates. Science, 367(6481), 984-986.

[3] Herbert, S., & Marculescu, D. (2007). Analysis of dynamic voltage/frequency scaling in chip-multiprocessors. In Proceedings of the 2007 international symposium on Low power electronics and design (ISLPED'07) (pp. 38-43).

[4] Liu, Z., Liu, I., Low. S., & Wierman, A. (2014). Pricing data center demand response. In 2014 ACM SIGMETRICS. (pp. 111-123).

[5] Liu, Z., Lin, M., Wierman, A., Low, S. H., & Andrew, L. L. (2011). Geographical load balancing with renewables. ACM SIGMETRICS Performance Evaluation Review, 39(3), 62-66.

[6] Lin, M., Liu, Z., Wierman, A., & Andrew, L. L. (2012). Online algorithms for geographical load balancing. In 2012 IEEE international green computing conference (IGCC) (pp. 1-10).

[7] Liu, Z., Wierman, A., Chen, Y., Razon, B., & Chen, N. (2013). Data center demand response: Avoiding the coincident peak via workload shifting and local generation. Performance Evaluation, 70(10), 770-791.

[8] Lei, H., Zhang, T., Liu, Y., Zha, Y., & Zhu, X. (2015). SGEESS: Smart green energy-efficient scheduling strategy with dynamic

electricity price for data center. Journal of Systems and Software, 108, 23-38.

[9] Niu, L., & Guo, Y. (2016). Enabling reliable data center demand response via aggregation. In Proceedings of the Seventh International Conference on Future Energy Systems (pp. 1-11).

[10] Yu, Z., Guo, Y., & Pan, M. (2017). Coalitional datacenter energy cost optimization in electricity markets. In Proceedings of the Eighth International Conference on Future Energy Systems (pp. 191-202).

[11] Zhou, Z., Liu, F., Chen, S., & Li, Z. (2018). A truthful and efficient incentive mechanism for demand response in green datacenters. IEEE Transactions on Parallel and Distributed Systems, 31(1), 1-15.

[12] Cupelli, L., Schütz, T., Jahangiri, P., Fuchs, M., Monti, A., & Müller, D. (2018). Data center control strategy for participation in demand response programs. IEEE Transactions on Industrial Informatics, 14(11), 5087-5099.

[13] Zhang, Y., Paschalidis, I. C., & Coskun, A. K. (2019). Data center participation in demand response programs with quality-of-service guarantees. In Proceedings of the Tenth ACM International Conference on Future Energy Systems (pp. 285-302).

[14] Tsiligkaridis, A., Paschalidis, I. C., & Coskun, A. (2019). Data center demand response pricing using inverse optimization. In Proceedings of the Tenth ACM International Conference on Future Energy Systems (pp. 400-402).

[15] Chen, H., Zhang, Y., Caramanis, M. C., & Coskun, A. K. (2019). Energyqare: Qos-aware data center participation in smart grid regulation service reserve provision. ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS), 4(1).

[16] Zhang, P., Li, K., Wang, F., Mi, Z., Chen, H., & Chang, S. (2020). A Cooperation Model of Data Center Cluster and Electricity Retailer Base on Demand Response. In 2020 IEEE/IAS 56th Industrial and Commercial Power Systems Technical Conference (I&CPS).

[17] Zhang, Y., Wilson, D. C., Paschalidis, I. C., & Coskun, A. (2021). HPC Data Center Participation in Demand Response: an Adaptive Policy with QoS Assurance. IEEE Transactions on Sustainable Computing.

[18] Niu, T., Hu, B., Xie, K., Pan, C., Jin, H., & Li, C. (2021). Spacial coordination between data centers and power system considering uncertainties of both source and load sides. International Journal of Electrical Power & Energy Systems, 124, 106358.

[19] Jahanshahi, A., Yu, N., & Wong, D. (2022). PowerMorph: QoS-aware Server Power Reshaping for Data Center Regulation Service. ACM Transactions on Architecture and Code Optimization (TACO).

[20] Tran, N. H., Tran, D. H., Ren, S., Han, Z., Huh, E. N., & Hong, C. S. (2015). How geo-distributed data centers do demand response: A game-theoretic approach. IEEE Transactions on Smart Grid, 7(2), 937-947.

[21] Hoefler, T., Jeannot, E., & Mercier, G. (2014). An overview of process mapping techniques and algorithms in high-performance computing. High Performance Computing on Complex Environments, 75-94.

[22] Singh, S., & Chana, I. (2016). A survey on resource scheduling in cloud computing: Issues and challenges. Journal of grid computing, 14(2), 217-264.

[23] Maiterth, M., Koenig, G., Pedretti, K., Jana, S., Bates, N., Borghesi, A., ... & Puzovic, M. (2018). Energy and power aware job scheduling and resource management: Global survey—initial analysis. In 2018 IPDPSW (pp. 685-693).

[24] Jin, C., de Supinski, B. R., Abramson, D., Poxon, H., ... & Jessup, E. R. (2017). A survey on software methods to improve the energy efficiency of parallel computing. The International Journal of High Performance Computing Applications, 31(6), 517-549.

[25] Cochran, R., Hankendi, C., Coskun, A. K., & Reda, S. (2011). Pack & cap: adaptive dvfs and thread packing under power caps. In 2011 IEEE 44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO) (pp. 175-185).

[26] Gholkar, N., Mueller, F., & Rountree, B. (2019). Uncore power scavenger: A runtime for uncore power conservation on hpc systems. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (pp. 1-23).

[27] Gu, C., Li, Z., Huang, H., & Jia, X. (2018). Energy efficient scheduling of servers with multi-sleep modes for cloud data center. IEEE Transactions on Cloud Computing, 8(3), 833-846.

[28] Sakamoto, R., Cao, T., Kondo, M., Inoue, K., Ueda, M., Patki, T., ... & Schulz, M. (2017). Production hardware overprovisioning: Real-world performance optimization using an extensible power-aware resource management framework. In 2017 IPDPS (pp. 957-966).

[29] Totoni, E., Jain, N., & Kale, L. V. (2015). Power management of extreme-scale networks with on/off links in runtime systems. ACM Transactions on Parallel Computing (TOPC), 1(2), 1-21.

[30] Zhou, Z., Liu, F., Zou, R., Liu, J., Xu, H., & Jin, H. (2015). Carbon-aware online control of geo-distributed cloud services. IEEE Transactions on Parallel and Distributed Systems, 27(9), 2506-2519.

[31] Parekh, A. K., & Gallager, R. G. (1993). A generalized processor sharing approach to flow control in integrated services networks: the single-node case. IEEE/ACM transactions on networking, 1(3), 344-357.

[32] Feitelson, D. G. (2001). Metrics for parallel job scheduling and their convergence. In Workshop on Job Scheduling Strategies for Parallel Processing (pp. 188-205). Springer, Berlin, Heidelberg.

[33] Paschalidis, I. C. (1999). Class-specific quality of service guarantees in multimedia communication networks. Automatica, 35(12), 1951-1968.

[34] Bertsimas, D., Paschalidis, I. C., & Tsitsiklis, J. N. (1999). Large deviations analysis of the generalized processor sharing policy. Queueing Systems, 32(4), 319-349.

[35] Bertsekas, D. P. (2016) Nonlinear Optimization. Athena Scientific.

[36] Laurie, D. et al. (2018). An open-source tool for controlling IPMI-enabled systems. Online: https://github.com/ipmitool/ipmitool.

[37] Bailey, D., Harris, T., Saphir, W., Van Der Wijngaart, R., Woo, A., & Yarrow, M. (1995). The NAS parallel benchmarks 2.0 (Vol. 156). Technical Report NAS-95-020, NASA Ames Research Center.

[38] Gholkar, N., Mueller, F., & Rountree, B. (2019). Uncore power scavenger: A runtime for uncore power conservation on hpc systems. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (pp. 1-23).

[39] Sakamoto, R., Cao, T., Kondo, M., Inoue, K., Ueda, M., Patki, T., ... & Schulz, M. (2017). Production hardware overprovisioning: Real-world performance optimization using an extensible power-aware resource management framework. In 2017 IPDPS (pp. 957-966).