Shaving Data Center Power Demand Peaks Through Energy Storage and Workload Shifting Control

Mehiar Dabbagh, Bechir Hamdaoui, Ammar Rayes[‡] and Mohsen Guizani[†]
Oregon State University, Corvallis, OR 97331, dabbaghm,hamdaoub@oregonstate.edu
[‡] Cisco Systems, San Jose, CA 95134, rayes@cisco.com
[†] University of Idaho, Moscow, ID 83844, mguizani@ieee.org

***Abstract*—This paper proposes efficient strategies that shave Data Centers (DCs)' monthly peak power demand with the aim of reducing the DCs' monthly expenses. Specifically, the proposed strategies allow to decide: i) when and how much of the DC's workload should be delayed given that the workload is made up of multiple classes where each class has a certain delay tolerance and delay cost, and ii) when and how much energy should be charged/discharged into DCs' batteries. We first consider the case where the DC's power demands throughout the whole billing cycle are known and present an optimal peak shaving control strategy for it. We then relax this assumption and propose an efficient control strategy for the case when (accurate/noisy) predictions of the DC's power demands are only known for short durations in the future. Several comparative studies based on real traces from a Google DC are conducted in order to validate the proposed techniques.**

***Index Terms*—energy efficiency, peak shaving, data centers, energy storage, workload shifting, convex optimization.**

I. Introduction

According to [1], large IT companies such as Google and Amazon spend millions of dollars per month on their Data Centers (DCs)' electricity bills. These bills account for 30% to 50% of the total DCs operational expenses [2]. Therefore, there is a great need for cutting down on those expenses. A DC's electricity bill is typically made up of two components [3]: i) *Energy Charge,* which is dependent on how much energy (in Kilo Watt Hour (KWH)) the DC consumes throughout the entire billing cycle (e.g. month), and ii) *Peak Charge*[1]*,* which is a penalty proportional to the maximum amount of power (measured in Kilo Watt (KW)) that was drawn by the DC during the billing cycle period. This penalty is very expensive and is enforced by the grid company to encourage the DC to balance its power demand and to discourage spiky power usages. The maximum amount of power drawn by the DC is calculated in a time-slotted fashion where the grid company calculates the DC's average power usage during each slot of a certain length (e.g. 15-minutes), and the peak charge is calculated based on the slot with the maximum average power among all the billing cycle's slots.

The majority of the prior techniques that were proposed to reduce the electricity bill focused exclusively on minimizing the Energy Charge while completely ignoring the Peak Charge. Unlike these techniques, this paper focuses on minimizing the Peak Charge component due to its high contribution to the total electricity bill as will be seen in our case studies. This is achieved by *shaving the peak* power drawn by the DC using a controller that tunes two knobs: 1) *Workload Shifting,* where some of the DC's computing jobs are delayed (and so are their corresponding power demands) during peak periods, and 2) *Energy Storage,* where extra power is drawn from the grid during low-demand periods and stored in batteries so that it can be used during peak periods later.

Challenges in finding efficient control strategies:

- *Battery Losses and Constraints:* Batteries are not ideal as they leak some of their stored energy over time; these losses are called *leakage losses*. In addition, due to conversion operations, energy can also be lost when it is being routed to the battery; these are called *conversion losses*. Furthermore, there is a limit on how much and how fast energy can be charged/discharged into batteries.
- *Workload Heterogeneity:* DC hosts jobs with different priorities and different Service-Level Agreements (SLAs). Each class of jobs can tolerate getting its requested power demands delayed by a certain amount of time and such delays have certain delay charges.
- *Workload Uncertainty:* DC workload changes over time and the duration of the billing cycle is long (typically one month). This makes it hard to make optimal control decisions as it is generally hard to know the DC's future power demands throughout the whole billing cycle.
- *Shaving Technique Selection:* Another key challenge is deciding which portion of the peak should be shaved by workload shifting and which one should be shaved by the stored energy based on multiple aspects such as delay tolerance, energy storage constraints (leakage/conversion losses, discharge/charge rate), and workload uncertainty.

This paper proposes a controller that considers all of the above-mentioned aspects. We start first by assuming that the DC's power demands throughout the whole billing cycle are known in advance (*full-horizon knowledge*), and we present a controller that finds the optimal control strategy for a battery with specific losses and constraints and for a workload with different classes in terms of delay tolerance and delay charges. The proposed full-horizon approach makes optimal decision when the DC's demands are predictable over the entire billing cycle. It also provides an upper bound of how much monetary

This work was supported in part by Cisco and NSF CAREER award CNS-0846044.
[1]Peak Charge is sometimes referred to by Demand Charge in the literature.

savings can be achieved and helps with selecting what type of battery the DC should be equipped with based on the DC's workload demands. We then consider the case where we only know the DC's power demand for a short duration in future (*limited-horizon knowledge*), where we propose a control algorithm that uses this knowledge to decide at each time step how much energy the battery needs to charge/discharge and whether the demanded power should be delayed for some of the classes of jobs hosted in the DC.

Main contributions of this paper:

- Proposed a peak shaving strategy that combines energy storage and workload shifting decisions to save energy.
- The strategy accounts for real energy storage losses and constraints, and considers a heterogeneous DC workload with multiple classes with each class having different delay tolerance and price.
- The strategy operates by solving a well formulated optimization problem that models and integrates the interactions among the different variables. The problem is proven to be convex and thus can be solved quickly.
- The strategy makes optimal decisions under full-knowledge of the future demands and significantly outperforms existing techniques under limited future knowledge. This was verified using real Google traces.

The rest is organized as follows. Section II illustrates why one should worry about reducing the DC's Peak Charge, and Section III introduces our notations. Sections IV and V present the full-horizon and limited-horizon control strategy frameworks. Section VI provides an evaluation study. Section VII summarizes how our work differs from prior work. Finally, Section VIII concludes and provides future directions.

II. Motivation: The Case for Google DC

In order to illustrate the significance contribution of the Peak Charge component to the total electricity bill, we conduct an experiment where we rely on real workload traces [4] from a Google DC that is made up of around 12K servers to calculate how much power that DC consumes over time. Google traces report the jobs that clients submitted to one of Google DCs. Each job belongs to one of four classes: non delay-sensitive, low delay-sensitive, medium delay-sensitive and high delay-sensitive[2]. We refer to these four classes by c_1, c_2, c_3 and c_4 respectively. Each submitted job is made up of a number of tasks where each task is assigned a Linux container and utilizes an amount of CPU resources over time. In order to calculate Google DC's power consumption, we parse the traces and track at each time slot how much CPU resources are being utilized by all the tasks belonging to all the jobs that are currently hosted on the Google DC. We then calculate for each time slot what is the least number of servers needed to be kept ON to server those tasks as if state-of-the-art Energy Charge minimizing techniques [5–8] were applied to consolidate the DC's workload. The power consumption of each ON server

[2]High delay-sensitive jobs are called production jobs in the traces and represent jobs that produce monetary revenue.

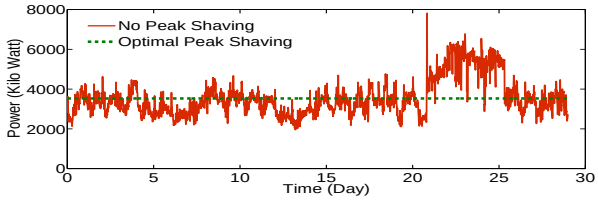


Fig. 1: The power drawn from Grid by Google DC.

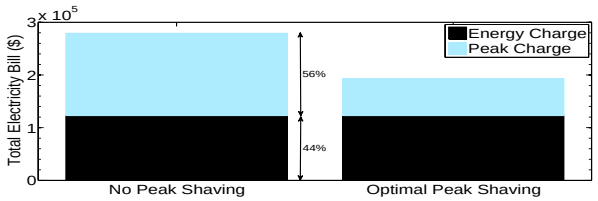


Fig. 2: Breakdown of Google DC total electricity bill (based on the power consumption in Fig. 1)

is then calculated based on the power model in [9] where the server's consumed power, P_{on} increases linearly from P_{idle} to P_{peak} as the server's CPU utilization, ν, increases from 0 to 100%. More specifically, $P_{on}(\nu) = P_{idle} + \nu(P_{peak} - P_{idle})$, where $P_{peak} = 400$ and $P_{idle} = 200$ Watts. The rest of the DC servers that are not hosting any tasks don't consume any power as they are assumed to be switched off completely or put to highly power efficient sleep states to save energy. Google DC is assumed to have a Power Usage Efficiency (PUE) of 1.7, which is a typical value for DCs [10] and means that for every watt spent on IT power, an additional 0.7 watt is spent by non-computing infrastructure (e.g. cooling devices).[3]

Fig. 1 plots the calculated power drawn by Google DC (referred to by No Peak Shaving) over the entire trace period (29 days). Fig. 1 also plots the power consumption of the Google DC when the same energy that was consumed by the DC during the 29-day period was spread evenly over the entire billing duration. This case is referred to by Optimal Peak Shaving as it represents the case where the DC had the same Energy Charge as the No Peak Shaving case but where the Peak Charge was minimal.

We then calculate the electricity bill for Google DC during the entire 29-day period[4] and using real power prices [11], where the price to calculate the Energy Charge is $\alpha = 0.05\,\$/KWH$, whereas the price to calculate the Peak Charge is $\beta = 20\,\$/KW$ and where the Peak Charge is calculated based on dividing the billing cycle into slots of length $\tau = 15$ minutes. We plot in Fig 2 the contribution of both the Energy Charge and the Peak Charge components to the total electricity bill based on the power consumption of No Peak Shaving

[3]It is worth mentioning that both the Energy Charge and the Peak Charge are directly proportional to the DC's PUE value. Thus the contribution (as a percentage of the total electricity bill) for the Peak Charge, the Energy Charge and the reported amount of savings that our approach achieves remain the same regardless of the DC's PUE value.

[4]Google revealed traces for only 29 days and thus our analysis considers the length of the billing cycle to be 29 days rather than 30 or 31 days.

and Optimal Peak Shaving cases that were shown in Fig. 1. Observe that for the No Peak Shaving case, the Peak Charge contributes to 56% of the total electricity bill while the Energy Charge accounts for the remaining 44%. Observe also that the Optimal Peak Shaving case reduces the paid Peak Charge during that month by \$86K when compared with the No Peak Shaving case, which is equivalent to a 30.8% reduction of the total electricity bill[5]. This translates into saving more than \$1M per year. These numbers highlight the high contribution of the Peak Charge to the total electricity bill and show the potentials for saving significant amount of money by making smart energy storage and workload shifting decisions.

III. Notations

We consider a time-slotted billing model, where the billing cycle is divided into n slots, each of length τ minutes. The index $i \in [1, \ldots, n]$ is used to refer to the billing cycle's slot i. All powers mentioned next are measured in Kilo Watt (KW). To easily distinguish the decision variables from the input parameters (the known quantities), small alphabetic letters are used exclusively to refer to the former whereas capital alphabetic letters or mathematical symbols (e.g. α, η) are used to refer to the latter. We introduce next the notations used in our paper (also summarized in Table I).

A. Energy Storage Notations

For a slot i, the power flow depicted in Fig. 3 uses the following notations:

- g_i: power taken from grid to serve the DC's power demands during the i^{th} slot.
- c_i^+: power taken from grid to charge battery in slot i.
- t_i: total power taken from the grid during the i^{th} slot.
- c_i^-: power discharged from the battery to serve the DC's power demands during the i^{th} slot.
- m_i: power that our controller decides to provide to the DC in slot i from both the grid and the battery. This power can be lower than the DC's power demands in slot i if the controller decides to delay providing the power demands of some of the DC's jobs. It can also be higher than the DC's power demands in slot i if our controller decides to provide power for some of the jobs that were delayed from the previous time slots.
- r_i: energy stored in battery at the beginning of slot i.

B. Battery Specs

The specs of the battery are summarized by the tuple $\Phi = (\eta_c, \eta_l, C_{max}^+, C_{max}^-, R_{max})$, where:

- η_c represents the battery's conversion efficiency and falls within the range $(0, 1]$, and means that only $\eta_c c_i^+$ percent out of the c_i^+ power that the battery draws from the grid ends up being stored in the battery, whereas the remaining $(1 - \eta_c)c_i^+$ gets lost due to conversion operations.

[5]We also estimated the savings that Optimal Peak Shaving achieves for a much lower peak price (β) values. As expected, the savings decrease as the peak price decreases but remain significant (23.8% for $\beta = 12\,\$/kw$ and 15.2% $\beta = 6\,\$/kw$).

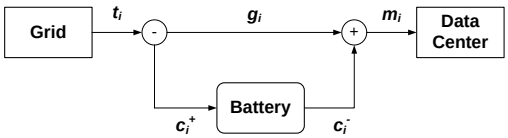


Fig. 3: Illustration of the power flow

- η_l represents the battery's leakage efficiency and falls within the range $(0, 1]$ and means that the battery losses $(1 - \eta_l)$ percent of its stored energy per one slot of time due to leakage losses.
- C_{max}^+ and C_{max}^- are the maximum charging and discharging rates which represent the maximum amount of power that the battery can draw from the grid and that the battery can discharge to the data center respectively.
- R_{max} represents the battery's maximum energy storage capacity that can be used for peak shaving.

In addition to those specs that are summarized by Φ, the initial energy that is stored in the battery at the beginning of the first slot in the billing cycle is referred to by R_{init}.

There are three things to mention regarding the battery model. First, although the term battery is normally used to refer to a device that stores energy chemically, it is actually used in this paper to refer to any energy-storage device regardless of the underlying storage technology as our battery model is general enough to cover the different technologies. Second, although the battery shown in Fig. 3 is represented as a single entity, it is actually made up of many storage cells that are normally placed in a large room in the DC's facility [2]. Finally, the DC's battery always stores a amount of energy called the backup energy that is enough to power the DC at full capacity for one minute while the remaining battery's capacity is used for peak shaving. The backup energy is used to power the DC during the transition time until the Diesel generator starts providing power when a power outage occurs. According to [12], this transition time is less than a minute while the DC's battery has usually enough capacity to power the DC at full capacity for more than half an hour.

C. Workload Shifting Notations

The DC draws power in order to execute jobs belonging to different classes. Let $\mathbb{C}$ be the set of all the job classes that the cluster hosts. Each class $c \in \mathbb{C}$ has a different priority and can tolerate getting the requested power demands within a *Delay Window* of length T^c time slots from the time the power demands are requested. The longer the length of the class's Delay Window, the less sensitive the jobs belonging to that class are to delays. We also consider the case where the DC is charged a delay cost for not providing the requested power demands directly. For a time slot i and a job class c, the following notations are used:

- D_i^c: power demanded by class c at time step i.
- $a_{i,k}^c$: power requested at time step i and allocated (provided) in time step k for class c where $i \leq k \leq i + T^c$.

- $l^c_{i,k}$: power left to provide at the beginning of time slot k from the power requested at slot i for class c.
- m^c_i: power provided for class c at step i. This power could be equal to D^c_i if all the class's requested power in slot i is directly provided with no delay and no power demands delayed from a previous slot were provided in slot i. The provided power m^c_i can also be less than D^c_i if a portion of the requested power demands in slot i is delayed for a latter slot. Also m^c_i can be greater than D^c_i if some of the class's power demands that were requested previously were deferred and were provided in slot i.

TABLE I: Summary of our notations.

Notation	Description
i	index of a slot in the billing cycle
g_i	grid power that the workload uses in slot i
c^+_i	power charged into the battery in slot i
t_i	total power drawn from grid in slot i
c^-_i	power discharged from the battery in slot i
m_i	power provided to the DC in slot i
r_i	energy left in the battery at the beginning of slot i
η_c	battery's conversion efficiency
η_l	battery's leakage efficiency
C^+_{max}	battery's maximum charging rate
C^-_{max}	battery's maximum discharging rate
R_{max}	battery's maximum storage capacity
Φ	a tuple that summarizes all the battery's specs
R_{init}	energy stored in battery prior to start of the billing cycle
$\mathbb{C}$	set of all job classes
c	index of one of the job classes
T^c	length of the delay window for class c
D^c_i	power demanded by class c at slot i
$a^c_{i,k}$	power requested at slot i but allocated at slot k for class c
$l^c_{i,k}$	power left to allocate at the beginning of slot k from the power requested at slot i for class c
m^c_i	power provided for class c at slot i
α	the energy price measured in ($\$/KWH$)
β	the peak price measured in ($\$/KW$)
γ^c_k	the cost for delaying the power demands of class c by k time slots measured in ($/KW)

IV. Full-Horizon Optimal Control

Given that we know the DC's power demands for each class $c \in \mathbb{C}$ throughout the whole billing period referred to by $\vec{D^c} = \{D^c_1, D^c_2, ..., D^c_n\}$, the specs of the battery Φ, and the initial amount of energy that the battery holds R_{init}, we find the optimal control strategy with the minimal electricity bill by solving the following optimization problem.

Objective: Minimize the total electricity bill made up of both Energy Charge and Peak Charge. We also seek to minimize Delay Charge that results from not providing the requested power demands directly for the different job classes. Thus, our objective can be expressed as:

$$\text{Minimize} \quad \underbrace{\alpha\xi \sum_{i=1}^{n} t_i}_{\text{Energy Charge}} + \underbrace{\beta \max_i\{t_i\}}_{\text{Peak Charge}} + \underbrace{\sum_{c\in\mathbb{C}} \sum_{k=1}^{T^c} \gamma^c_k \sum_{i=1}^{n} a^c_{i,i+k}}_{\text{Delay Charge}}$$

where α is the energy price in $\$/KWH$, $\xi = \tau/60$ is a constant that converts the total energy the DC draws from the grid into KWH, β is the peak price in $\$/KW$, and γ^c_k is the cost in \$/KW of delaying the power demands of class c by k time slots.

Constraints: The optimization problem is solved subject to the following constraints. One,

$$t_i = g_i + c^+_i \qquad , 1 \le i \le n \tag{C.1}$$

which states that for each time slot i the total power drawn from the grid is the aggregation of the power used to sever the DC's power demand and the power drawn to be stored in the battery. Two,

$$g_i + c^-_i = m_i \qquad , 1 \le i \le n \tag{C.2}$$

which states that the power provided to the DC is taken from the grid and from the power discharged from the battery for each time slot i. Three,

$$r_i = \begin{cases} R_{init} & , i = 1 \\ \eta_c c^+_{i-1}\tau + \eta_l(r_{i-1} - c^-_{i-1}\tau) & , 1 < i \le n \end{cases} \tag{C.3}$$

Which calculates how much energy will be stored in the battery at the beginning of each slot while accounting for the conversion and leakage losses. Four,

$$c^-_i \tau \le r_i \qquad , 1 \le i \le n \tag{C.4}$$

which states that the amount of discharged energy within the slot i that has a duration of τ must not exceed the amount of energy that is stored in the battery. Five,

$$c^-_i \le C^-_{max} \qquad , 1 \le i \le n \tag{C.5}$$

which states that the discharged power at any time slot must not exceed the maximum discharging rate that the battery supports. Six,

$$c^+_i \le C^+_{max} \qquad , 1 \le i \le n \tag{C.6}$$

which states that the charged power at any time slot must not exceed the maximum supported charging rate. Seven,

$$r_i \le R_{max} \qquad , 1 \le i \le n \tag{C.7}$$

which states that the amount of energy that the battery stores is bounded by the battery's maximum storage capacity. Eight,

$$m_i = \sum_{c\in\mathbb{C}} m^c_i \qquad , 1 \le i \le n \tag{C.8}$$

which states that the power provided to the DC at time slot i is the aggregation of the power provided for each class $c \in \mathbb{C}$ at that time slot. Nine,

$$t_i, g_i, c^+_i, c^-_i, r_i, m_i \ge 0 \qquad , 1 \le i \le n \tag{C.9}$$

which states that these decision variables are all non-negative.

Recall that each class c has a certain delay window T^c within which the requested power demands must be provided.

Thus, for each class $c \in \mathbb{C}$ the following constraints (C.10 to C.15) must hold:

$$m_i^c = \sum_{k=max\{i-T^c,\,1\}}^{i} a_{k,i}^c \qquad ,1 \leq i \leq n \tag{C.10}$$

Which states that the power provided for class c at time slot i is the aggregation of the power that was request at i and provided directly without delay in addition to the power deferred from previous time slots. We take $k = max\{i-T^c,\,1\}$ as there are no power demands prior to the first time slot.

$$l_{i,i+1}^c = D_i^c - a_{i,i}^c \qquad ,1 \leq i \leq n \tag{C.11}$$

Which states that the power left to allocate at the beginning of time slot $i+1$ is equal to the amount of power that was requested and wasn't allocated directly at time slot i.

$$l_{i,k+1}^c = l_{i,k}^c - a_{i,k}^c \qquad ,1 \leq i \leq n,\ i+1 \leq k \leq i+T^c \tag{C.12}$$

Which states that the power left to allocate at time slot $k+1$ is equal to the amount of power that was left to allocate at the beginning of the previous time slot k and that wasn't allocated within time slot k.

$$l_{i,i+T^c+1}^c = 0 \qquad ,1 \leq i \leq n \tag{C.13}$$

Which prevents the power demand for class c from being delayed more than the delay window T^c.

In order to restrict our decisions to one billing cycle, we add the following constraint which prevents deferring the power demands that are at the end of the billing cycle (which we are trying to optimize) to the following billing cycle:

$$l_{i,n+1}^c = 0 \qquad ,n - T^c \leq i \leq n \tag{C.14}$$

This constraint can be relaxed if we are optimizing over multiple billing cycles. Finally,

$$m_i^c, a_{i,j}^c, l_{i,k}^c \geq 0 \tag{C.15}$$

which states that these decisions variables are non-negative and this holds $1 \leq i \leq n$, $i \leq j \leq i+T^c$, $i+1 \leq k \leq i+T^c$.

The formulated problem is a convex optimization problem [13] as the objective is a convex function that we seek to minimize, all equality constraint functions are affine, and all non-equality constraints are convex functions. The solution of convex problems can be found quickly and there are well-developed tools that can be used to calculate the optimal solution efficiently such as the CVX package [14], which is the one used in our implementation.

Problem Formulation Generalization: Our formulation can also capture the following additional aspects:

- Battery Life Cycles: The battery's Depth of Discharge (DoD) can be constrainted not to exceed a certain percentage (P_{DoD}) in order to guarantee that the battery lives for a certain number of cycles as follows:

$$r_i \geq (1 - P_{DoD})R_{max}, 1 < i \leq n \tag{C.16}$$

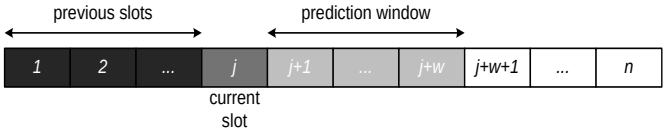


Fig. 4: Temporal illustration of the prediction window.

- Varying Energy Prices: The energy price α was set to be constant, but nothing actually prevents it from being different from one time slot to another within the same billing cycle (due to fluctuations in the energy market). In fact, this variability allows our controller to use the battery to store energy when the energy price is low so that it can be used later when the energy price is high in addition to the peak shaving usage. This is again done while taking into account the battery's specs (leakage and conversion losses, etc.).
- Dropping Power Demands: In addition to storing energy and shifting power demands for different workload classes, dropping the demanded powers (partially or fully) can be added as another action that our controller can take. This can be achieved by adding $\sum_{c\in\mathbb{C}}\sum_{i=1}^{n}\lambda^c d_i^c$ to the objective that we seek to minimize, where λ^c is the cost in \$/KW for dropping the demanded power for class c and d_i^c is the amount of power in KW that our controller decides to drop at the i^{th} slot for class c. An additional constraint $d_i^c \geq 0$, $1 \leq i \leq n$, should be also added to indicate that the dropped power can't be negative, and constraint C.11 needs to be replaced by $l_{i,i+1}^c = D_i^c - a_{i,i}^c - d_i^c$, $1 \leq i \leq n$, as the dropped power at slot i needs to be excluded from the power left to allocate at the beginning of slot $i+1$ for class c.

V. Limited-Horizon Control

We discussed previously how to find the optimal control strategy when the DC's power demands throughout the whole billing cycle are known. We now consider the case where we only know predictions[6] of the DC's power demands in a short duration in the future (referred to by the Prediction Window) and we propose an algorithm that uses these predictions in order to make energy storage and workload shifting decisions at each time slot while accounting for the battery's energy losses and for the different workload class's delay tolerance and delay costs. A pseudo code of our proposed algorithm (Algorithm1) is presented to better illustrate our algorithm and Fig. 4 provides a temporal illustration of the slots that we will refer to while explaining the pseudo code of our algorithm. The pseudo code of our algorithm gets launched at the beginning of each slot j and takes the following inputs:

- j, index of current slot for which energy storage and workload shifting decisions need to be made.

[6]The predictions are calculated dynamically based on the recent demands and the latest decisions. This can be achieved using workload predcition techniques such as those we proposed in [8] and [9].

- $\vec{D_j}$, a vector that holds the power demanded for each class $c \in \mathbb{C}$ at the j^{th} slot (the current slot). This vector can be expressed as $\vec{D_j} = \{D_j^{c_1}, D_j^{c_2}, \ldots, D_j^{c_{|\mathbb{C}|}}\}$
- Φ, battery specs (see Section III.B).
- w, prediction window length, which represents the number of slots in the future for which the DC's power demands need to be predicted.
- t_{max}, maximum amount of total power drawn from the grid so far up to slot j, and can be calculated[7] as $\max_{1 \leq k < j}\{t_k\}$ when $j > 1$ and 0 when $j = 1$.

As illustrated in the pseudo code, for each workload class, our algorithm starts first by predicting the class's power demands in upcoming w slots (Line 2), where these predicted power demands are referred to by $\hat{D}^c_{j+1}, \hat{D}^c_{j+2}, \ldots, \hat{D}^c_{j+w}$. These predicted demands can be obtained using any machine learning technique that provides accurate predictions. The focus of this paper is on how to use these predictions to make energy storage and workload shifting control decisions and not on what technique to use to obtain accurate predictions. Our algorithm also fetches for each class the amount of power deferred from the previous slots (Line 3), where the notation $L^c_{i,j}$ is used to refer to class c's power demands that were requested at time slot i and that were not provided up to the beginning of the j^{th} slot based on our algorithm's decisions when it was launched in the previous slots. Recall that for class c the power demands that are requested at slot i must be provided within the following $i + T^c$ slots. Thus there are no power demands delayed from more than $j - T^c$ previous slots for class c as our algorithm ensures that these demanded power gets allocated within their Delay Window.

Our algorithm then fetches r_j which represents the amount of available stored energy in the battery at the beginning of the j^{th} slot (basically the amount of energy left in the battery after conversion and leakage losses). Now in order to determine the best control actions, our algorithm solves in Line 6 an optimization problem called the Limited-Horizon Optimization Problem (LHOP) that is similar to the full-horizon optimization problem (Section IV) but with the following key differences. First, LHOP seeks to minimize the electricity and delay costs by considering only the predicted power demands from slot j to slot $j + w$ while also keeping an eye on the maximum amount of power t_{max} that was drawn from the grid in the previous slots of the billing cycle. The max power demand in the previous slots is needed as the Peak Charge is calculated based on the slot with the maximum power drawn from the grid throughout the whole billing cycle. Thus LHOP basically considers the maximum power drawn from the grid from the the first slot in the billing cycle and up to the $j + w$ slots. The slots beyond $j + w$ are not considered by LHOP as it is hard to predict the DC's power demands for more than w slots. The constraints (C'.1 to C'.15) in LHOP are the equivalent of constraints (C.1 to C.15) that were explained before for the full optimization problem with the exceptions that the LHOP constraints consider only the decision variables involved in the period from slot j and up to $j+w$ and that the control decisions need to meet the predicted power demands rather than the actual power demands in the $j + w$ future slot (constraint C'.11). The last key different that distinguish LHOP is that it has three additional constraints (C'16 to C'18) that are included to guarantee that the leftover power demands from the previous slots are provided within the current or future slots and before the end of their Delay Window (i.e., the power demands requested at slot $j - i$ must be provided within $j - i + T^c$ slots).

LHOP is a convex optimization problem [13] as the objective is a convex function that we seek to minimize, all equality constraint functions are affine, and all non-equality constraints are convex functions. Solving LHOP returns the best control decisions that need to be made in the period from slot j and up to slot $j + w$. Our algorithm then commits only to the control decisions in the j^{th} slot (Line 7) that are returned by solving LHOP, while the control decisions in each of the following slots will be determined later when the pseudo code of our algorithm is launched again at the beginning of each one of those following slots.

[7]It is also possible to set t_{max} initially (at $j = 1$) to a value that is larger than zero and that is known to be below the average DC's power demands. This prevents our algorithm from shaving the short peaks that are below the t_{max} value that might be encountered at the beginning of the billing cycle and that are way below the peak encountered in the whole billing cycle. This is not considered in our evaluations as it requires some knowledge about the characteristics of the DC's power demands.

VI. Evaluation

We rely on the real power prices and on the power demands for Google DC that were introduced in Section II, and we conduct comparative experiments to evaluate how much money Google DC ends up saving in a one billing cycle when using our peak shaving control strategy. Our evaluations are organized into three subsections:

A. Peak Shaving through Energy Storage

In this subsection, we consider the case where the DC's workload is not allowed to be shifted (i.e. , $T^c = 0 \quad \forall c \in \mathbb{C}$). We study the savings that our controller achieves when making only energy storage decisions when Google DC is supplied by each of the following energy storage technologies:

- **Lead-Acid (LA):** this battery uses electrochemistry to store and to discharge energy.
- **Lithium-Ion (LI):** relies also on electrochemistry but uses different chemical components where the cathode is a lithiated metal oxide and the anode is a graphite carbon.
- **Ultra-Capacitors (UC):** uses a double layer electrochemistry to store energy between the electrodes.
- **Fly-Wheels (FW):** is a mechanical energy storage device that uses the momentum of a wheel/cylinder to store energy.
- **Optimal (OPT):** represents an ideal battery that has zero conversion and leakage losses and unlimited charging/discharging rate.

The first four types represent the most popular energy storage technologies that are found in DCs [11], whereas the OPT

Algorithm 1 LimitedHorizonControl$\big(\ j,\ \vec{D}_j,\ \Phi,\ w,\ t_{max}\ \big)$

1: **for each** $c \in \mathbb{C}$ **do**

2: $\overbrace{[\hat{D}^c_{j+1},\ \hat{D}^c_{j+2},\ \ldots,\ \hat{D}^c_{j+w}]}^{\text{Predicted Power Demands}} \leftarrow$ predictFutureDemands(w)

3: $\overbrace{[L^c_{j-T^c,j},\ \ldots,\ L^c_{j-2,j},\ L^c_{j-1,j}]}^{\text{Previous Leftovers}} \leftarrow$ getPreviousLeftovers( )

4: **end for**

5: $R_j \leftarrow$ getAmountOfStoredEnergy()

6: Solve Limited-Horizon Optimization Problem (LHOP):

Minimize

$$\underbrace{\alpha\xi \sum_{i=j}^{j+w} t_i}_{\text{Energy Charge}} + \underbrace{\beta max\{\max_{j\le i\le j+w}\{t_i\}, t_{max}\}}_{\text{Peak Charge}} + \underbrace{\sum_{c\in\mathbb{C}}\sum_{k=1}^{T^c}\gamma^c_k \sum_{i=j-T^c}^{j+w} a^c_{i,i+k}}_{\text{Delay Charge}}$$

subject to

$$t_i = g_i + c^+_i \quad ,j \le i \le j+w \quad \text{(C'.1)}$$

$$g_i + c^-_i = m_i \quad ,j \le i \le j+w \quad \text{(C'.2)}$$

$$r_i = \begin{cases} R_j & ,i = j \\ \eta_c c^+_{i-1}\tau + \eta_l(r_{i-1} - c^-_{i-1}\tau) & ,j < i \le j+w \end{cases} \quad \text{(C'.3)}$$

$$c^-_i\tau \le r_i \quad ,j \le i \le j+w \quad \text{(C'.4)}$$

$$c^-_i \le C^-_{max} \quad ,j \le i \le j+w \quad \text{(C'.5)}$$

$$c^+_i \le C^+_{max} \quad ,j \le i \le j+w \quad \text{(C'.6)}$$

$$r_i \le R_{max} \quad ,j \le i \le j+w \quad \text{(C'.7)}$$

$$m_i = \textstyle\sum_{c\in\mathbb{C}} m^c_i \quad ,j \le i \le j+w \quad \text{(C'.8)}$$

$$t_i, g_i, c^+_i, c^-_i, r_i, m_i \ge 0 \quad ,j \le i \le j+w \quad \text{(C'.9)}$$

for each $c \in \mathbb{C}$

$$m^c_i = \textstyle\sum_{k=max\{i-T^c,1\}}^{i} a^c_{k,i} \quad ,j \le i \le j+w \quad \text{(C'.10)}$$

$$l^c_{i,i+1} = \begin{cases} D^c_i - a^c_{i,i} & i = j \\ \hat{D}^c_i - a^c_{i,i} & j < i \le j+w \end{cases} \quad \text{(C'.11)}$$

$$l^c_{i,k+1} = l^c_{i,k} - a^c_{i,k} \quad ,j \le i \le j+w \quad ,i+1 \le k \le i+T^c \quad \text{(C'.12)}$$

$$l^c_{i,i+T^c+1} = 0 \quad ,j \le i \le j+w \quad \text{(C'.13)}$$

$$l^c_{i,n+1} = 0 \quad ,j+w-T^c \le i \le j+w \quad \text{(C'.14)}$$

$$m^c_i, a^c_{i,j}, l^c_{i,k} \ge 0 \quad \text{(C'.15)}$$

$$l^c_{j-i,j+1} = L^c_{j-i,j} - a^c_{j-i,j} \quad ,1 \le i \le T^c \quad \text{(C'.16)}$$

$$l^c_{j-i,j+k} = l^c_{j-i,j+k-1} - a^c_{j-i,j+l-1}, 1 \le i \le T^c, 1 < k \le T^c - i + 1 \quad \text{(C'.17)}$$

$$l^c_{j-i,j-i+T^c+1} = 0 \quad ,1 \le i \le T^c \quad \text{(C'.18)}$$

end for

7: Make Control Actions Specified by $(t_j,\ g_j,\ c^+_j,\ c^-_j,\ m_j,\ m^c_j)$

battery represents an unrealistic ideal battery. The specs of these batteries are presented in Table II and are based on [11]. We also assume that there is no stored energy initially at the beginning of the billing cycle when evaluating the different types of batteries (i.e., $R_{init} = 0$). These battery specs will be used throughout the paper unless otherwise specified.

TABLE II: Specs for the considered battery types [11].

	LA	LI	UC	FW	OPT
Conversion Efficiency (%)	75	85	95	95	100
Leakage Efficiency (% per day)	99.7	99.9	80	1	100
Max Charging Rate (mega Watt)	16	16	8	8	∞
Max Discharging Rate (mega Watt)	8	8	8	8	∞
Storage Capacity (mega Watt hour)	16	16	16	16	∞

1) Full-Horizon Control Evaluation: Fig. 5 shows the total electricity bill for running the Google DC for a one billing cycle under different scenarios, where the total bill is broken down into the Energy Charge and the Peak Charge for each scenario. The "No Peak Shaving" scenario represents the case when the DC's power demands are drawn only from the grid without using any battery for peak shaving. The other scenarios in Fig. 5 show the total electricity bill for Google DC when different types of batteries were used to shave the peak where each type of those batteries is operating based on the decisions of the full-horizon controller that was proposed in Section IV. The results clearly highlight that the DC's total electricity bill can be reduced significantly if our proposed full-horizon control technique was used to control how much energy needs to be charged/discharged over time for the different types of batteries. Observe that the total electricity bill is lower for the LI and LA battery types when compared to the FW and UC as the former types have lower leakage losses than the latter types, which allows storing larger amount of energy to be used to shave the peak that is encountered later, without leaking much of their stored energy over time. The Energy Charge of the FW, UC, LI and LA batteries are slightly higher than those of "No Peak Shaving" due to the leakage and conversion losses which increase the amount of energy that the DC consumes over time. However, these extra Energy Charge leads into significant reduction in the Peak Charge which leads in turn into significant reduction of the total electricity bill.

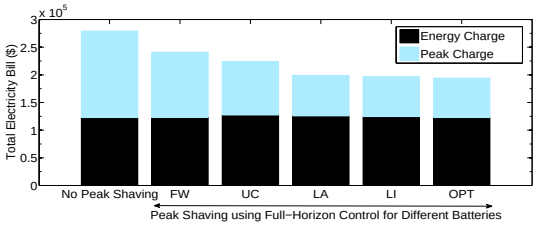


Fig. 5: Full-horizon control monetary savings for the different types of batteries based on Google traces.

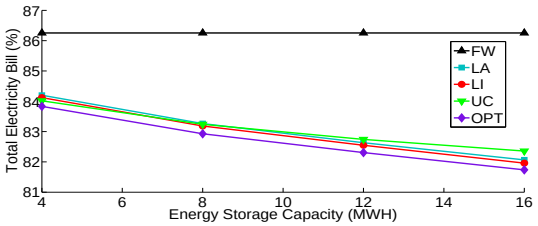


Fig. 6: Google DC total electricity bill for different battery technologies that are operating by our proposed full-horizon controller under different energy storage capacities.

We vary next the energy storage capacity R_{max} for each type of battery and we report in Fig. 6 the total electricity bill of Google DC (normalized w.r.t. No Peak Shaving total costs) when the proposed full-horizon controller is making charging/discharging decisions. As expected, for each battery type the larger the energy storage capacity, the higher the amount of power that can be shaved and the lower the total electricity bill. Observe that increasing the energy storage capacity for the FW energy storage device causes a negligible

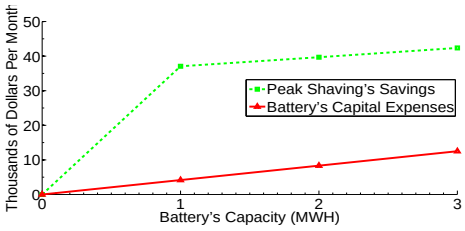


Fig. 7: Battery's capital expenses and peak shaving savings under different capacity for LA battery.

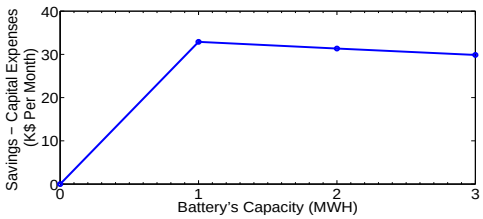


Fig. 8: Difference between the capital expenses and the peak shaving's savings under different capacity for LA battery.

additional reductions in the electricity bill as this energy storage technology is highly leaky. This shows that from a peak shaving perspective and for the Google traces, an FW energy storage device with a capacity of 4 MWH is as effective as one with a 16 MWH capacity. The former requires less facility space and has a lower capital costs and thus would be a better choice when considering the energy storage capacity of the battery that the DC should be equipped with. In fact, the monetary savings in Fig. 6 together with the capital expenses and the facility space limitation can be used to decide what battery technology and what energy storage capacity the DC should be equipped with.

4

To further illustrate the effect that the capacity of the battery has on the capital expenses and on the peak shaving savings, we relied on the cost estimates in [15], which state that the capital expenses for a LA battery are 50 $/KWH/year, and plotted in Fig. 7 the battery's capital expenses (shown in red solid line) per month as the battery's capacity was increased. We plotted then the peak shaving savings that our full-horizon controller achieves per month as a function of the battery's capacity (shown in green dashed line). As could be seen, the larger the capacity of the battery, the higher the amount of peak that can be shaved, and the larger the monthly electricity bill savings. Observe from Fig. 7 that the savings that peak shaving achieves under these capacity values are higher than the battery's capital expenses, which shows that the money spent on buying a battery for peak shaving will be paid off. We plotted next in Fig. 8 the difference between the monthly peak shaving savings and the monthly battery's capital expenses. Observe that based on Fig. 8, it is more cost effective to use a battery with a capacity of 1 MWH than a battery with a larger capacity as the increase in the battery's capital expenses after that point outweighs the extra savings. Such analysis can help in deciding the capacity of the battery that the DC should be equipped with based on the DC's power demands and the battery's specification (e.g. leakage and conversion losses, etc.). We would like to mention that DCs are normally equipped with batteries for fault-tolerance and so the battery's capital expenses shown in Fig 8 in that case are inevitable regardless of whether or not the DC is using the battery for peak shaving. Of course using the battery for peak shaving besides its regular use for fault tolerance would increase slightly the inevitable capital expenses as the battery in that case needs to be replaced more often. Finally, we would like to mention that the results reported in Fig. 7 and 8 show the savings that our controller achieves when using a single knob (energy storage) for peak shaving. Our approach can also use a second knob (workload shifting) to achieve higher savings and this knob does not increase the capital expenses.

2) Limited-Horizon Control Evaluation: Fig. 9 reports the total electricity bill associated with running Google DC for a one billing cycle when different controllers are used to make charging/discharging decisions for the different types of batteries. More specifically, for each battery type, Figure 9 reports the electricity for each of the following controllers:

- **No Peak Shaving:** represents the case where all the DC's power demands are provided solely from the grid where the battery is not charging4/discharging any energy.
- **Full-Horizon:** represents the controller proposed in Section IV which makes the optimal control decisions for each battery type as it has full-knowledge of the DC's demands within the whole billing cycle.
- **Limited-Horizon (Oracle):** is the control algorithm that we proposed in Section V when operating under 100% accurate predictions of the DC's power demand in each of the following four slots (i.e., $w = 4$).
- **Limited-Horizon (Noisy):** is similar to the previous case with the exception that a random noise drawn from a Gaussian distribution with zero mean and a standard deviation of 200 kilo Watts is added to the predicted power demand that are provided to our algorithm in each time step. The added noise represents prediction errors and can take either a positive or negative value to mimic over or under estimation of the predicted power demands.
- **Threshold:** is a well-known technique [16–18] that compares the DC's demanded power at each slot against a fixed threshold. If the demanded power is below the threshold, then the power difference is charged into the battery within that slot. Otherwise, the battery discharges the difference (or whatever amount of energy less than the difference that is stored in the battery). The charging (discharging) power is capped s.t. it doesn't exceed the battery's maximum charging (discharging) rate and s.t. no extra energy beyond the battery's capacity is being charged. Tuning the threshold for each battery type is done by evaluating different threshold values on one-day traces where the value that yielded the least electricity expenses is picked for evaluation.

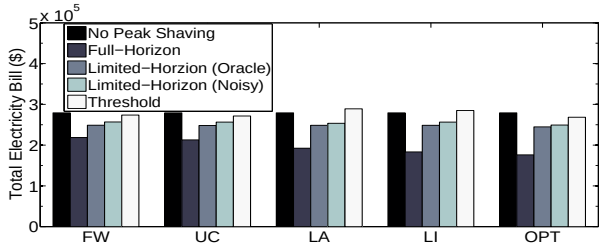


Fig. 9: Comparative evaluation of the different control techniques for each battery type based on Google traces.

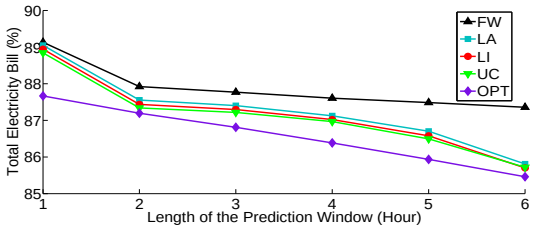


Fig. 10: The electricity bill of Google DC as the length of the Prediction Window increases for the Limited-Horizon (Oracle) controller when operating different types of batteries.

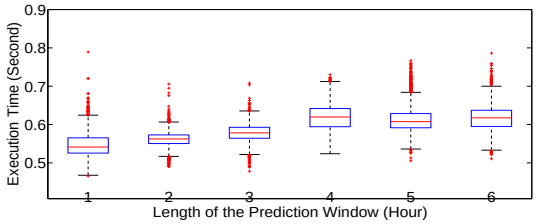


Fig. 11: The time that the Limited-Horizon controller took to make decisions for different lengths of Prediction Window.

Observe from Fig. 9 that for each type of battery the Limited-Horizon control (both Oracle and Nosy cases) had a lower total electricity bill than the No Peak Shaving case and than the Threshold controller. As expected, the Noisy case had higher electricity bill than the Oracle case due to the added prediction errors that affected slightly the decisions of our algorithm. Obviously the total electricity bill for the Limited-Horizon control is higher than that of the Full-Horizon control as the latter has the advantage of knowing the DC's power demand throughout the whole billing cycle which allows it to make the optimal battery control decisions. Notice that the gap between the Full-Horizon and the Limited-Horizon controller is small for FW when compared to the remaining battery types. This is attributed to the fact that FW is highly leaky (has low leakage efficiency and hence high leakage losses), thus the amount of energy that FW stores decays quickly over time which leaves only a small energy that can be used for peak shaving in the future slots that are far in time. Thus for highly leaky batteries (such as FW), knowing the future power demands for only a short duration in the future for the Limited-Horizon controller makes reductions in the electricity bill that are close to the Full-Horizon (optimal) case. These results are when the Limited-Horizon Algorithm relied only on predictions of the DC's power demands in the following four slots. Recall that each slot has a duration of 15 minutes and thus this represents the case where our algorithm knows the predicted power demands in the next hour.

We increased next the length of the Prediction Window and plotted in Fig. 10 the total electricity bill for Google DC (normalized w.r.t. No Peak Shaving total electricity costs) when the Limited-Horizon (Oracle) controller is making charging and discharging decisions for each battery type. The results clearly show that for each battery type, the larger the length of the Prediction Window, the higher the peak power that can be shaved, and hence the lower the total electricity bill. For a certain length of Prediction Window, the gap between the OPT battery and any other battery technology is attributed to the leakage and conversion losses (as the OPT battery does not have any leakage or conversion losses). It is worth mentioning that for the FW energy storage device, increasing the length of the Prediction Window leads into very small additional reductions as this battery is highly leaky and thus only a small percentage of the drawn energy at any time slot will remain in the battery to use in the slots that are far in time. It is also worth mentioning that for all types of batteries, in the extreme case when the length of the Prediction Window of the Limited-Horizon (Oracle) controller is equal to the length of the billing cycle, then the Limited-Horizon (controller) performs exactly as the Full-Horizon controller.

We investigated next in Fig. 11 how much time it took the Limited-Horizon controller to solve the optimization problem and to make charging/discharging decisions under different Prediction Window lengths. Since an optimization problem needs to be solved at the beginning of each time slot in order to make the appropriate power decisions, we track for each slot how much time it took to solve the optimization problem and then we show bar plots for these execution time under each Prediction Window. The measured times are calculated based on running the Matlab code of our controller on a machine that has a CPU frequency of 2.6 Ghz and a 62 GB RAM. The variability for each Prediction Window in Fig. 11 is attributed to the fact that the inputs (e.g. the power demands) vary from an optimization problem into an other which affect the time needed to find the problem's optimal solution. Observe that increasing the length of the Prediction Window does increase the time needed to make control decisions as the optimization problem has now larger number of variables that need to be found. However, it took always very small amount of time (less than a second) to solve the optimization problem in all of those cases which is something vital for online DC management.

In order to illustrate the decisions that our controllers make, we plot in Fig. 12 the power demands of Google DC over time and we plot in a different color the total grid power (the aggregation of the power provided to the DC and the power charged into the battery) for the Full-Horizon and for the Limited-Horizon (Oracle) controllers when operating an LA battery. The power demanded by the DC at any point in

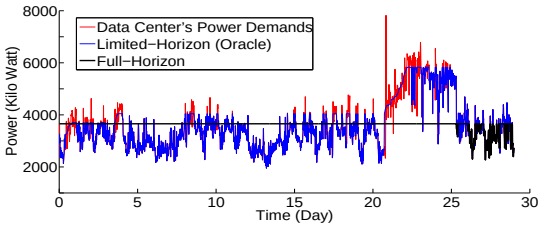


Fig. 12: The power demands of Google DC and the total power drawn from the grid by the full-horizon and the limited-horizon (Oracle) controllers for the LA battery.

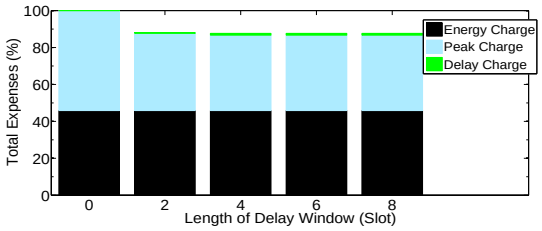


Fig. 13: Google DC's expenses when the Full-Horizon controller is making workload shifting decisions for different Delay Window length.

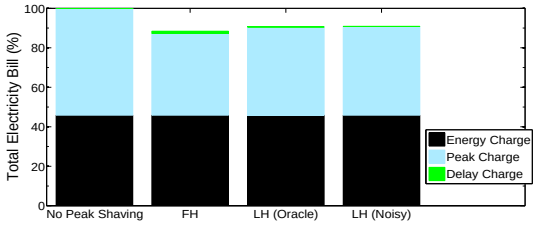


Fig. 14: Google DC's total expenses when only the power demands for the low and non delay-sensitive jobs are allowed to be delayed.

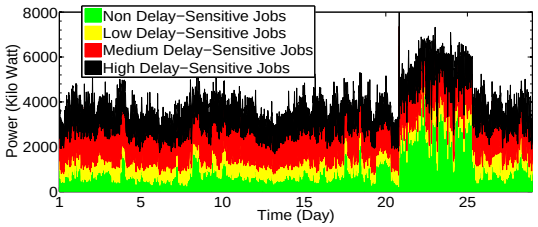
time is provided by our controllers. This means that each time the total grid power for any of those controllers goes above the DC's demands then the difference is being routed to be stored in the battery. Also each time the total grid power is below the DC's demands then the difference is drawn from the battery to supply the DC's demands. Fig. 12 clearly shows that the DC's peak power demand at the 21st day was shaved significantly by each of our controllers. This reduced the Peak Charge and hence minimized the DC's electricity bill.

B. Peak Shaving through Workload Shifting

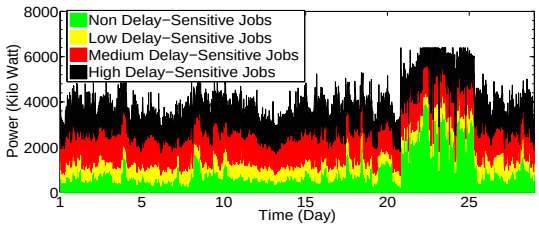
Recall that the jobs that are reported in Google traces are classified into four classes: c_1, c_2, c_3 and c_4 where the class with a lower index contains less delay-sensitive jobs. In this subsection, we consider the case where the DC is not supplied by a battery that can store energy for peak shaving purposes (i.e., R_{max} = 0) but where the power demands of some of the jobs belonging to certain classes are allowed to be delayed for a certain duration. Throughout the paper, the costs for delaying the demanded power is made linearly proportional to the number of slots for which the requested power demands were delayed for each class. More specifically, $\gamma_k^c = k\lambda^c$ where γ_k^c is the cost for delaying 1 KW of the power demands of class c for k time slots and λ^c is the cost for delaying 1 KW of the power demands of class c for one time slot. We evaluate next both the Full-Horizon and the Limited-Horizon controllers when making workload shifting decisions.

1) Full-Horizon Control Evaluation: We consider first the case where only the power demands of the jobs belonging to classes c_1 and c_2 (the least delay-sensitive jobs) can be delayed to be provided within a Delay Window of length T^{c_1} slots and T^{c_2} slots respectively and with costs $\lambda^{c_1} = 0.01$ \$/KW and $\lambda^{c_2} = 0.02$ \$/KW respectively. In this experiment, we consider the case where: $T^{c_1} = T^{c_2}$ (i.e. , both classes have the same Delay Window length). We then increase the length of the Delay Window for those two classes and we plot in Fig. 13 the total expenses of Google DC (normalized w.r.t. "No Peak Shaving" expenses) when our proposed Full-Horizon controller is making workload shifting decisions. For each case, the total expenses are broken down into the Energy Charge, the Peak Charge and the Delay Charge. Observe that for the case when the Delay Window has a zero length, there is no Delay Charge as our controller is basically forced to provide the requested power demands directly with no delay which resulted in total expenses that are the same as the "No Peak Shaving" case. Observe that as the length of the Delay Window increases, the Peak Charge decreases and so does the DC's total expenses as our controller gains larger ability to delay the requested power demands over longer periods, which increases the amount of power that can be shaved. Notice that the Energy Charge for the different cases is the same as in all those cases the energy consumption (in KWH) was identical as our controller provides all the requested power demands (either directly or after some delay).

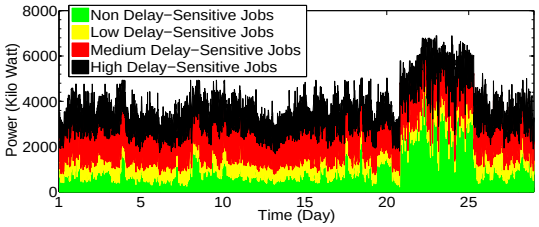
2) Limited-Horizon Control Evaluation: We consider the case where classes c_1 and c_2 are allowed to be delayed but when each one of those classes has a different Delay Window length where $T^{c_1} = 3$ slots and $T^{c_1} = 1$ slot. We show in Fig. 14 Google DC's total expenses (normalized w.r.t. No Peak Shaving) and broken down into the Energy, Peak and Delay charge components for the Full-Horizon (FH) controller as well as for the Limited-Horizon (Oracle) and the Limited-Horizon (Noisy) controllers when knowing predictions of the future power demands in the following hour. Observe that the LH (Oracle) controller had a total expenses that are only 3% larger than the FH controller. This shows that knowing the future power demands for a short duration of one hour in the future achieves very close reductions to those of knowing the power demands throughout the whole billing cycle. As expected, the LH (Noisy) controller had slightly higher total expenses than the LH (Oracle) controller due to the prediction errors which affected the decisions made by this controller. Fig. 15 further illustrates how the Peak Charge was reduced by the Full-Horizon and the Limited-Horizon (Oracle) controller where we show how the power demands for the four classes were allocated over time for the "No



(a) No Peak Shaving.



(b) Full-Horizon Workload Shifting.



(c) Limited-Horizon (Oracle) Workload Shifting.

Fig. 15: Breakdown of Google power consumption for the different job classes.

Peak Shaving" case and for the Full-Horizon and the Limited-Horizon (Oracle) controllers. Observe that the Full-Horizon and Limited-Horizon controllers avoided the peak at the 21st day by delaying the power demands of the Non delay-sensitive and low delay-sensitive classes and spreading them over the following slots. This reduced the Peak Charge and lead into significant reductions in the the DC's total expenses.

C. Peak Shaving through Energy Storage & Workload Shifting

1) Full-Horizon Control Evaluation: We evaluate first in Fig. 16 our Full-Horizon controller where we compare Google DC's total expenses when making only Workload Shifting decisions (referred to by WS), when making only Energy Storage decisions (referred to by ES), and when making both Energy Storage and Workload Shifting decisions (referred to by ES+WS). The results clearly show that our controller makes the most reductions in the total expenses when making both Energy Storage and Workload Shifting decisions as our controller stores energy during the low power periods (the valleys) that are before the peak power demand and then delays some of the power demands within the peak periods to a latter time that has a low demand. Observe that the Delay Charge for the (ES+WS) case is lower than the (WS) as using the battery to store energy for peak shaving purposes reduces the time needed to delay the workload in order to shave the peak which in turn reduces the Delay Charge.

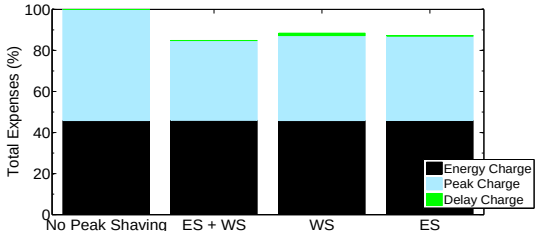


Fig. 16: Google DC's total expenses when our Full-Horizon controller is making only Workload Shifting, only Energy Storage and both Energy Storage & Workload Shifting decisions.

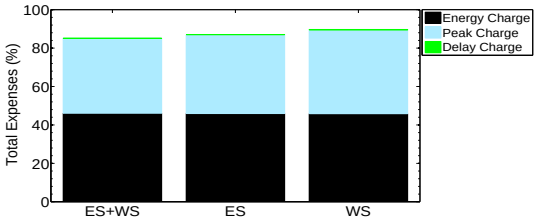


Fig. 17: Google DC's total expenses (normalized w.r.t. "No Peak Shaving" expenses) when our Limited-Horizon controller is making only Workload Shifting, only Energy Storage and both Energy Storage & Workload Shifting decisions.

2) Limited-Horizon Control Evaluation: In our final experiment, we evaluate in Fig. 17 our Limited Horizon controller when making both Energy Storage and Workload Shifting decisions (referred to by ES+WS), when making only Energy Storage (ES), and when making only Workload Shifting (WS) decisions. The results clearly show that by controlling the two knobs: storing energy and workload shifting, our controller was able to achieve more reductions in the expenses than the remaining cases.

The results in Fig. 16 and Fig. 17 are when the DC is supplied by an LA battery that has the following specs: $C^+_{max} = C^-_{max} = 8MW$, $R_{max} = 8MWH$ and leakage and conversion efficiency as specified in Table II. Only the non delay-sensitive jobs (class c_1) and the low delay-sensitive jobs (class c_2) were allowed to be delayed within these experiments where $T^{c_1} = 3$ slots, $T^{c_2} = 1$ slot, $\lambda^{c_1} = 0.02$ \$/KW and $\lambda^{c_2} = 0.04$ \$/KW. The evaluations in Fig. 17 for the Limited-Horizon controller are when knowing one day ahead predictions of the future power demands.

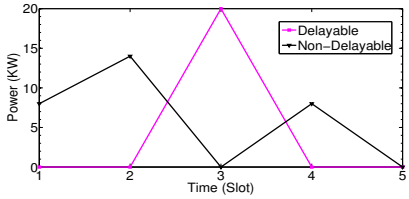


Fig. 18: Requested power in our example.

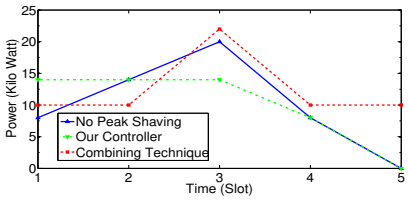


Fig. 19: Power drawn from grid under different controllers.

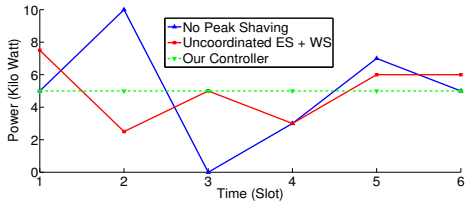


Fig. 20: Power drawn from grid under different controllers.

D. Comparison Against a Simple Combining Technique

The example provided in this subsection illustrates that using a simple controller that combines both workload shifting and storing energy would lead into suboptimal decisions. We consider the case where the DC's battery is optimal and where there are two types of workloads: i) *Non-Delayable* and ii) *Delayable* for the next time slot with no delay charge. Fig. 18 illustrates the power demands of the Non-Delayable and Delayable workloads over the few time slots that respresent our billing cycle in our example. We consider a combining technique that compares the aggregate power demands of the DC against a fixed threshold (10 KW) at each time slot. If the demands are below the threshold, then the difference is charged into the battery. Whereas if the demands are above the threshold, then the battery is first used to provide the difference. If no enough energy is available in the battery to provide the whole difference, then workload shifting is used to shave the remaining amount that is above the threshold. We illustrate in Fig. 19 the power drawn from the grid to satisfy the aggregate power demands of our example under: a) No Peak Shaving, b) the previously described combining technique, and c) under our limited-horizon controller that knows the DC's power demand in the next two slots. Observe that the combining technique increased the peak by (10%) compared to No Peak Shaving as it delays the workload when it meets a high demand (without considering if there will be a higher demand in the following time slot) and then finds itself forced to meet the deadline leading into a higher peak. The combining technique also continues to charge energy into the battery whenever the demand is below the threshold which increases the energy charge. Our controller on the other hand reduced the peak by (30%) compared to No Peak Shaving.

E. Comparison Against Uncoordinated Combining Technique

We consider in this example the case where the DC's battery is optimal and where all the power demands can be delayed to the next slot with no charge. The solid blue line in Fig. 20 shows the DC's power demands in our example without peak shaving. The red line represents the case when applying both workoad shifting and energy storage to shave the peak, but without coordination. This means that the workload shifting controller is unaware of the decisions made by the energy storage controller and vice versa. However, at each point in time, the energy storage controller and the workload shifting controller know accurate predictions of the DC's power demands in the next two slots. Each of these two controller makes the optimal peak shaving decisions separately by solving an optimization problem. Observe that for this case, the peak gets reduced by 25% compared to 'No Peak Shaving'. Finally, the green dashed line shows the drawn power when our controller is making both energy storage and workload shifting decisions in a coordinated fashion (i.e., the decisions made to store energy affect those made to shift the workload and vice versa). Observe that the peak gets reduced in this case by 50% compared to 'No Peak Shaving', which is double the reductions achieved by the uncoordinated case.

VII. Related Work

A. Energy Charge Minimization Techniques

Energy-aware scheduling [5, 6], Virtual Machine (VM) migration [7] and server over-booking [8] are some of the techniques that were proposed to minimize the DC's electricity bill. All of these techniques reduce the Energy Charge component of the electricity bill by using fewer number of ON servers. The main limitation of these techniques [5–8] is that unlike our work they completely ignore the Peak Charge.

B. Energy Storage Peak Shaving Techniques

Reinforcement learning was used in [19] to make battery charging/discharging decisions based only on the DC's current power demand and the amount of stored energy. The main limitation of this technique is that it requires tuning some parameters (e.g., learning rate) empirically, and discretizes the input and output spaces into a finite set, leading to less accurate decisions. Markov Decision Process (MDP) was also used to derive peak sharing control strategies [20]. Unlike our work, this technique does not target minimizing Peak Charges. Threshold-based techniques (e.g., [16, 18]) aimed to operate the DC at a fixed power by charging the difference between a fixed threshold and the DC's power demands into the battery or by discharging from the battery the difference between the DC's power demand and the threshold. Bounds on the performance of this technique were derived in [17] using the concept of arrival curves from Network Calculus. Although simple, these approaches are threshold sensitive. The authors in [21, 22] applied the Threshold technique on multi-hierarchical levels of a DC that has a distributed UPS topology where each server is attached to an independent battery. The work in [23] analyzed the different architectures and battery technologies that can be used for such a distributed UPS topology. Dynamic Programming (DP) was also proposed in [24] to find the optimal control strategy for a battery with no losses under full-knowledge of the future power demands.

Unlike the DP approach, our proposed full-horizon controller solves a well-formulated convex optimization problem that considers the battery's losses and power constraints. Liu et al. investigated in [25] the benefits obtained from integrating hybrid energy storage technologies into DCs and proposed an efficient management scheme for a DC equipped with both ultra capacitors and conventional UPS systems. The management scheme in [25] is actually complementary to our work as it can be used in the case of multiple energy storage technologies to decide the percentage of charge/discharge for each energy storage technology while our work can run on top of that scheme to decide the total amount of power that needs to be charged/discharged by all the DC equipped energy storage technologies in addition to the amount of workload that needs to be delayed (shifted) to avoid the peak penalties. Finally, a key distinction of our controller from all these existing techniques is that it considers workload shifting as an additional control knob to achieve further peak shavings.

C. Workload Modulation Peak Shaving Techniques

The Peak Charge was reduced in [26, 27] by dropping some of the DC's requested or scheduled jobs within periods with high power demands. Clients in these cases clearly experience service disruptions as their dropped jobs are required to be resubmitted and to start again from scratch. The work in [28, 29] explored redirecting/migrating jobs from the DC with high peak power demands into other DCs that have lower demands in order to reduce the Peak Charge. The authors in [30] capped the power consumed by the DC's servers in addition to performing inter DC migrations. The migration approaches [28–30] are complementary to our work as they can be applied on top of our controller to balance the workload among multiple DCs. The authors in [31] considered using both local power generation and workload shifting to reduce the peak charges. However, the main limitation of their workload shifting algorithm is that it is non-adaptive as once decisions are made at the beginning of each period, they can't be changed within that period no matter how the workload fluctuates. This makes the algorithm's solution sub optimal. Another key difference that distinguishes our work from the work in [31] is the fact that we consider storing energy as another knob to shave the peak. The work in [32] considered delaying and dropping jobs for peak shaving. Our paper differs from [32] in that we consider the case where the DC's workload is divided into multiple classes where each class has a different delay tolerance and a different delay cost. Another key difference from [32] is that we control two knobs simultaneously workload shifting and energy storage. To the best of our knowledge, our proposed peak shaving controller is the first that adjusts both of these knobs. In order to shave the peak demands, Zheng et al. [33] considered using Thermal Energy Storage (TES) tanks to make iced water during low electricity price periods so that it can be used later to reduce the power spent by the cooling infrastructure during peak power demand periods. Unlike the work in [33], our work is not limited only to DCs equipped with TES tanks. The authors in [34] considered the specific case where the DC is equipped with fuel cells and suggested a controller that uses both energy storage devices and power capping to address the fact that fuel cells react slowly to power surges. Unlike [34], our controller is not specific to fuel cells. Another key difference from [33] and [34] is that we use both energy storage and workload shifting to shave the DC's peak charges.

VIII. Conclusion and Future Work

We propose peak shaving strategies that minimize the DC's electricity bill by making smart energy storage and workload shifting decisions. Our proposed strategies solve a well-formulated convex optimization problem that considers: real battery's specs such as leakage and conversion losses, maximum charging/discharging rate and storage capacity, heterogeneous workload classes, each having different delay tolerance and delay charge. Using real Google traces, we show that our controllers achieve promising reductions in DCs' electricity bills. For future work, we plan to develop accurate workload prediction techniques and to further study other factors in our control strategy such as the battery's lifetime.

References

[1] J. Koomey, "Growth in data center electricity use 2005 to 2010," *A report by Analytical Press, completed at the request of The New York Times*, 2011.

[2] L. Barroso, J. Clidaras, and U. Hölzle, "The datacenter as a computer: An introduction to the design of warehouse-scale machines," *Synthesis lectures on computer architecture*, vol. 8, no. 3, pp. 1–154, 2013.

[3] C. Wang, B. Urgaonkar, G. Kesidis, U. Shanbhag, and Q. Wang, "A case for virtualizing the electric utility in cloud data centers," in *Proceedings of USENIX conference on Hot Topics in Cloud Computing*, 2014.

[4] C. Reiss, J. Wilkes, and J. Hellerstein, "Google cluster-usage traces: format+ schema," *Google Inc., White Paper*, 2011.

[5] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Exploiting task elasticity and price heterogeneity for maximizing cloud computing profits," *IEEE Transactions on Emerging Topics in Computing*, vol. PP, no. 99, pp. 1–1, 2015.

[6] M. Shojafar, N. Cordeschi, D. Amendola, and E. Baccarelli, "Energy-saving adaptive computing and traffic engineering for real-time-service data centers," in *IEEE Communication Workshop (ICCW)*, 2015.

[7] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Towards energy-efficient cloud computing: Prediction, consolidation, and overcommitment," *IEEE Network Magazine*, 2015.

[8] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "An energy-efficient vm prediction and migration framework for overcommitted clouds," *IEEE Transactions on Cloud Computing*, 2016.

[9] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Energy-efficient resource allocation and provisioning framework for cloud data centers," *IEEE Transactions on Network and Service Management*, vol. 12, no. 3, pp. 377–391, 2015.

[10] M. Ghamkhari et al., "Optimal integration of renewable energy resources in data centers with behind-the-meter renewable generator," in *International Conference on Communications (ICC)*, 2012.

[11] D. Wang, C. Ren, A. Sivasubramaniam, B. Urgaonkar, and H. Fathy, "Energy storage in datacenters: what, where, and how much?," in *ACM SIGMETRICS Performance Evaluation Review*, 2012, pp. 187–198.

[12] R. Urgaonkar, B. Urgaonkar, M. Neely, and A. Sivasubramaniam, "Optimal power cost management using stored energy in data centers," in *ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, 2011, pp. 221–232.

[13] S. Boyd and L. Vandenberghe, "Convex optimization," *Cambridge University Press*, 2004.

[14] Inc. CVX Research, "CVX: Matlab software for disciplined convex programming, version 2.0," 2012.

[15] D. Wang, S. Govindan, A. Sivasubramaniam, A. Kansal, J. Liu, and B. Khessib, "Underprovisioning backup power infrastructure for datacenters," in *ACM SIGARCH Computer Architecture News*, 2014, vol. 42.
[16] M. Johnson, A. Bar-Noy, O. Liu, and Y. Feng, "Energy peak shaving with local storage," *Sustainable Computing: Informatics and Systems*, vol. 1, no. 3, pp. 177–188, 2011.
[17] W. Munawar and C. Jian-Jia, "Peak power demand analysis and reduction by using battery buffers for monotonic controllers," in *International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, 2013.
[18] A. Bar-Noy, M. Johnson, and O. Liu, "Peak shaving through resource buffering," in *International Workshop on Approximation and Online Algorithms*, 2008, pp. 147–159.
[19] Y. Wang and M. Pedram, "Model-free reinforcement learning and bayesian classification in system-level power management," *IEEE Transactions on Computers*, 2016.
[20] P. van de Ven, N. Hegde, L. Massoulié, and T. Salonidis, "Optimal control of end-user energy storage," *IEEE Transactions on Smart Grid*, vol. 4, no. 2, pp. 789–797, 2013.
[21] V. Kontorinis et al., "Managing distributed ups energy for effective power capping in data centers," in *IEEE International Symposium on Computer Architecture (ISCA)*, 2012, pp. 488–499.
[22] B. Aksanli, T. Rosing, and E. Pettis, "Distributed battery control for peak power shaving in datacenters," in *IEEE Green Computing Conference (IGCC)*, 2013, pp. 1–8.
[23] V. Kontorinis, L. Zhang, B. Aksanli, J. Sampson, H. Homayoun, E. Pettis, D. Tullsen, and T. Rosing, "Managing distributed ups energy for effective power capping in data centers," in *Annual International Symposium on Computer Architecture (ISCA)*, 2012, pp. 488–499.
[24] A. Oudalov, R. Cherkaoui, and A. Beguin, "Sizing and optimal operation of battery energy storage system for peak shaving application," in *IEEE Power Tech.*, 2007, pp. 621–625.
[25] L. Liu, C. Li, H. Sun, Y. Hu, J. Gu, T. Li, J. Xin, and N. Zheng, "HEB: deploying and managing hybrid energy buffers for improving datacenter efficiency and economy," in *ACM SIGARCH Computer Architecture News*, 2015, vol. 43, pp. 463–475.
[26] H. Xu and B. Li, "Reducing electricity demand charge for data centers with partial execution," in *ACM Proceedings of the international conference on Future energy systems*, 2014, pp. 51–61.
[27] N. Chen, X. Ren, S. Ren, and A. Wierman, "Greening multi-tenant data center demand response," *Performance Evaluation Journal*, vol. 91, pp. 229–254, 2015.
[28] R. Wang, N. Kandasamy, C. Nwankpa, and D. Kaeli, "Datacenters as controllable load resources in the electricity market," in *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2013, pp. 176–185.
[29] M. Etinski, M. Martonosi, K. Le, R. Bianchini, and T. Nguyen, "Optimizing the use of request distribution and stored energy for cost reduction in multi-site internet services," in *Sustainable Internet and ICT for Sustainability (SustainIT)*, 2012, pp. 1–10.
[30] S. Govindan, D. Wang, A. Sivasubramaniam, and B. Urgaonkar, "Aggressive datacenter power provisioning with batteries," *ACM Transactions on Computer Systems (TOCS)*, vol. 31, no. 1, pp. 2, 2013.
[31] Z. Liu, A. Wierman, Y. Chen, B. Razon, and N. Chen, "Data center demand response: Avoiding the coincident peak via workload shifting and local generation," *Performance Evaluation*, vol. 70, no. 10, pp. 770–791, 2013.
[32] C. Wang, B. Urgaonkar, Q. Wang, G. Kesidis, and A. Sivasubramaniam, "Data center power cost optimization via workload modulation," in *International Conference on Utility and Cloud Computing*, 2013.
[33] W. Zheng, K. Ma, and X. Wang, "Exploiting thermal energy storage to reduce data center capital and operating expenses," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2014, pp. 132–141.
[34] Y. Li et al., "Sizecap: Efficiently handling power surges in fuel cell powered data centers," in *High Performance Computer Architecture*, 2016.

Mehiar Dabbagh received the PhD degree from Oregon State University in 2016, the MS degree from the American University of Beirut in 2010 and the BS degree from the University of Aleppo in 2005, all in ECE. During his PhD studies, he interned with Cisco and with HP where he worked on developing cloud-related technologies. His research interests include: Cloud Computing, Distributed Systems, Energy Efficiency and Data Mining.

Bechir Hamdaoui (S'02-M'05-SM'12) is presently an Associate Professor in the School of EECS at Oregon State University. He received the Diploma of Graduate Engineer (1997) from the National School of Engineers at Tunis, Tunisia. He also received M.S. degrees in both ECE (2002) and CS (2004), and the Ph.D. degree in Computer Engineering (2005) all from the University of Wisconsin-Madison. His current research focus is on distributed optimization, parallel computing, cognitive networking, cloud computing, and Internet of Things.

Ammar Rayes is a Distinguished Engineer at Cisco Systems and the Founding President of The International Society of Service Innovation Professionals, www.issip.org. He is currently chairing Cisco Services Research Program. His research areas include: Smart Services, Internet of Everything (IoE), Machine-to-Machine and IP strategy. He received his BS and MS Degrees in EE from the University of Illinois at Urbana and his PhD in EE from Washington University in St. Louis, Missouri.

Mohsen Guizani (S'85-M'89-SM'99-F'09) is currently a Professor at the University of Idaho. He also served in academic positions at the University of Missouri-Kansas City, University of Colorado-Boulder, Syracuse University and Kuwait University. He received his B.S. (with distinction) and M.S. degrees in EE; M.S. and Ph.D. degrees in CS in 1984, 1986, 1987, and 1990, respectively, all from Syracuse University, Syracuse, New York. His research interests include Wireless Communications, Computer Networks and Cloud Computing.