Dynamic Scheduling and Pricing in Wireless Cloud Computing

Shaolei Ren and Mihaela van der Schaar

Abstract—In this paper, we consider a wireless cloud computing system in which the service provider operates a data center and provides cloud services to its subscribers at dynamic prices. We propose a joint optimization of scheduling and pricing decisions for delay-tolerant batch services to maximize the service provider's long-term profit. Unlike the existing research on jointly scheduling and pricing that focuses on static or asymptotic analysis, we focus on a dynamic setting and develop a provably-efficient Dynamic Scheduling and Pricing (Dyn-SP) algorithm which, without the necessity of predicting the future information, can be applied to an arbitrarily random environment that may follow an arbitrary trajectory over time. We prove that, compared to the optimal offline algorithm with future information, Dyn-SP produces a close-to-optimal average profit while bounding the job queue length in the data center. We perform a trace-based simulation study to validate Dyn-SP. In particular, we show both analytically and numerically that a desired tradeoff between the profit and queueing delay can be obtained by appropriately tuning the control parameter. Our results also indicate that, compared to the existing algorithms which neglect demand-side management, cooling system energy consumption, and/or the queue length information, Dyn-SP achieves a higher average profit while incurring (almost) the same average queueing delay.

Index Terms—Cloud computing, pricing, profit maximization, scheduling, stochastic optimization

1 Introduction

Enabled by ubiquitous communications and low-cost mobile devices, the proliferation of digital data and growing demand for outsourcing the data to a cloud for processing has attracted a significant amount of attention from various IT companies, which advocate wireless cloud computing as one of the future gold mines [1]. While dynamically providing computing resources (e.g., turning servers on/off [2]) has been praised as an effective approach to reducing cloud operation costs and thereby increasing the service provider's profit, recent studies show that combining dynamic resource provisioning with other operations such as load balancing can further enhance the system performance in terms of delay and energy consumption [3], giving the service provider an edge in the cloud computing market.

In this paper, we explore the interaction of *dynamic* pricing and service request scheduling, and demonstrate that jointly optimizing these two decisions can significantly enhance the wireless cloud service provider's *long-term* profitability. Towards this end, we focus on a wireless service provider operating multiple base stations and one data center. Wireless subscribers *outsource* data-intensive and/or computing-intensive applications to the cloud managed by the service provider for processing. Migrating such applications is mutually beneficial: users can access a large pool of high-performance computing resource anywhere and anytime; the service provider is no longer only providing wireless connectivity, but can enhance its profit by integrating cloud computing with wireless access services. Example applications that can, or even have to, be migrated to the cloud include real-time stream mining, scientific computing, visual search, and batch image processing [4]. In general, these applications can be classified as *interactive* (i.e., delay-sensitive) and *batch* (i.e., delay-insensitive), and recent studies show that batch workloads typically account for 70% of all workloads in data centers [5].[1] Focusing on *dynamically* scheduling and pricing for batch services that exhibit a high degree of scheduling flexibility due to the delay-tolerant nature, we aim at developing an efficient online algorithm to maximize the service provider's long-term profit in a *random* environment. This is in stark contrast to the existing research on joint optimization of scheduling and pricing that focuses on a limiting regime (e.g., when the capacity goes to infinity) [8] or a static game-theoretic setting for demand response in smart grid [9], [10]. Developing efficient online algorithms is a nontrivial task: the random environment[2] that is difficult

- *S. Ren is with the School of Computing and Information Sciences, Florida International University, Miami, FL 33199 USA. E-mail: renshaolei@gmail.com.*
- *M. van der Schaar is with the Electrical Engineering Department, University of California, Los Angeles (UCLA), Los Angeles, CA 90095 USA. E-mail: mihaela@ee.ucla.edu.*

Manuscript received 23 Oct. 2012; revised 14 Apr. 2013; accepted 25 Apr. 2013. Date of publication 1 May 2013; date of current version 26 Aug. 2014. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier 10.1109/TMC.2013.57

1. Real-world traces show that batch workloads can take up more than 50% of all data center workloads, as can be seen by comparing the power consumption of batch and interactive services in [6]. Thus, we take the liberty of foreseeing that batch services will be an important ingredient in the era of wireless cloud computing, as corroborated by recent studies [7].

2. We use the term *environment* to collectively refer to on-site renewable energy generation, electricity price, server availability, as well as wireless network capacity.

to accurately predict in the long term impedes the derivation of the optimal offline scheduling and pricing decisions. While hour-ahead information of the environment is readily available in practice [11], [12], accurately predicting the long-term information for *all* these parameters are very challenging, if not impossible.[3]

To tackle the randomness in the environment, we propose a provably-efficient online algorithm, Dynamic Scheduling and Pricing (Dyn-SP), which can be implemented using the currently available information without the necessity of predicting the future. Dyn-SP builds upon the Lyapunov optimization technique that only requires a very mild condition which almost all the practical scenarios satisfy [13]. We rigorously prove that Dyn-SP is efficient in the sense that the profit gap between the Dyn-SP and the optimal offline algorithm with T-slot future information is upper bounded. Meanwhile, the job queue length is also upper bounded, resulting in a finite (and bounded) average queueing delay. By appropriately adjusting a control parameter (denoted by V), the tradeoff between the queueing delay and profit can be flexibly adjusted. We also derive performance bounds of Dyn-SP by explicitly considering monotonically increasing control parameters of V that may be used in practical scenarios. We conduct extensive simulations to validate Dyn-SP. In particular, we show that the long-term profit achieved by Dyn-SP can be pushed arbitrarily close to the maximum (at the expense of increasing the queueing delay). Compared to other scheduling algorithms which neglect pricing, Dyn-SP produces a higher profit while incurring the same queueing delay, thereby highlighting the importance of jointly considering scheduling and pricing in a random environment.

The rest of this paper is organized as follows. Related work is reviewed in Section 2. Section 3 describes the model. In Section 4, we develop a provably-efficient online algorithm to maximize the service provider's average profit subject to queueing delay constraint. Simulation results are shown in Section 5 and finally, concluding remarks are offered in Section 6.

2 Related Work

In this section, we provide a snapshot of the existing research from the following aspects.

• **Wireless cloud computing:** With a plethora of data and mobile applications, providing cloud-based services to wireless subscribers has attracted special attention from various wireless carriers (e.g., [1]), enabling new business opportunities. For instance, [14] proposes a new architecture migrating data-intensive and/or computation-intensive applications from mobile devices to the cloud, whereas [7], [15] advocate the migration of (some or part of) applications from energy-constrained mobile devices to the cloud.

• **Energy-efficient data center operation:** There has been a growing interest in optimizing data center operation by cutting electricity bills and reducing energy consumption [11], [16]–[20]. In particular, "power proportionality" via dynamically turning on/off servers based on the workloads (a.k.a. dynamic capacity provisioning or right-sizing) has been recently studied and advocated as a promising approach to reducing data center energy costs [2], [18]. Combined with dynamic capacity provisioning, geographical load balancing can potentially provide additional benefits in cost saving [19], [20], response time reduction [12], and reducing the brown energy usage (e.g., by "following the renewables" or scheduling the workloads to data centers with more green energies) [11].

• **Pricing:** Pricing has been widely used in engineering as a lever to enable efficient resource management in wireless networks (e.g., [21], [22]), congestion control (e.g., [23], [24]), technology adoption in communication markets [25], and demand coordination in smart grid [9]. In cloud computing, [26] proposes a pricing algorithm to regulate admission and perform resource allocation in such a way that the social welfare is maximized, and [27] develops a computationally efficient pricing mechanism to enable fair competition for cloud resource. Recently, there has been a surging interest in *dynamic* pricing in smart grid and wireless networking. For instance, various demand-responsive pricing schemes have been proposed to coordinate the real-time energy consumption: [9] presents an autonomous and distributed demand-side energy management system among self-interested users; by considering consumers' device-specific scheduling flexibility (e.g., dish washers can be scheduled to late night, whereas air conditioners must be activated as requested), [10] develops an optimal day-ahead pricing scheme for the electricity provider to minimize its cost. In wireless networking, a notable example of applying dynamic pricing is that, building upon the theoretic framework of utilizing dynamic pricing for network congestion management [24], a new system architecture as well as a user trial of time-dependent pricing (TDP, equivalent to dynamic pricing in our study) has been successfully implemented [28], benefiting both wireless operators (increasing profit and flattening traffic demand) and customers (saving month bills).

To our best knowledge, joint optimization of scheduling and pricing is relatively less explored in cloud computing. While such an optimization paradigm has been considered in other domains (e.g., manufacturing management [8] and smart grid [9], [10]), our study fundamentally differs from the existing research in that we consider *dynamic* optimization to maximize the service provider's long-term objective in a *random* environment: we develop a probably-efficient online algorithm that is applicable for a random environment whose dynamics can follow an almost arbitrary trajectory over time. Although some of the existing studies have considered online scheduling (but without considering pricing) for data center optimization, they rely on the assumption that the environment follows i.i.d. distribution or a Markov chain [18], [29], [30], which, however, does not hold well in practice as shown by several studies [31]. Other online algorithms that are applicable for almost arbitrary environments (like our study, but without considering pricing) use *competitive ratio* as the worst-case performance analysis [12]. Nonetheless, these online algorithms are difficult to incorporate pricing, because the complex demand function that may not even be convex [32]. Last

3. A more detailed discussion of the random environment is provided in the supplementary material, which is available in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TMC.2013.57.

TABLE 1
List of Notations

Notation	Description
$b(t)$	Batch service demand
$d(t)$	Processed batch jobs
$f(t)$	Cooling system energy consumption
$y(t)$	Renewable energy supply
$p(t)$	Price for batch service
$u_i(b(t),t)$	Time-dependent utility for representative user i
$\phi(t)$	Electricity price
$r(t)$	Electricity cost
$h(t)$	Profit
$q(t)$	Batch job queue length
$C_i(t)$	Network capacity of base station i

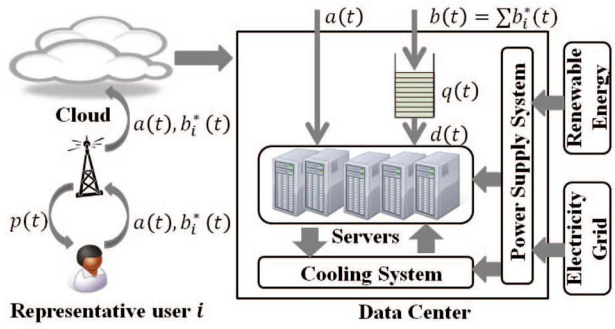

Fig. 1. System diagram.

but not least, we derive performance bounds of Dyn-SP by explicitly considering monotonically increasing control parameters of V that may be used in practical scenarios.

3 Model

We consider a discrete-time model in which the duration of each time slot matches the timescale of prediction window for which the service provider can *accurately* predict the future information. For example, if the service provider can only predict the hour-ahead future information, then each time slot corresponds to one hour and the service provider can update its scheduling and pricing decisions at the beginning of each hour. Nevertheless, if the service provider is able to perform a longer-term prediction (e.g., day-ahead prediction), then each time slot corresponds to the prediction window, at the beginning of which the service provider needs to select (a sequence of) decisions that will be used throughout the prediction window. In the following analysis, we mainly focus on hour-ahead prediction for the convenience of presentation, while noting that the model applies as well to longer-term prediction. Next, we describe the service provider model as well as the user model, and then discuss the extension of our model. Key notations are listed in Table 1.

3.1 Service Provider

As illustrated in Fig. 1, we consider a wireless service provider that provides cloud computing services to its subscribers. To keep the model succinct, we concentrate on delay-tolerant batch services while deferring the discussion of delay-sensitive interactive services to the extended model.

3.1.1 Base Station

The service provider owns N base stations, indexed by $1, 2, \ldots, N$, respectively, each of which covers a certain area.[4] We use $C_i(t) \in [C_{\min}, C_{\max}]$ to represent the network capacity available for batch services provided by base station i at time t. The network capacity is time-varying for various reasons such as the users' distances to the respective base station. The maximum amount of batch jobs that can be submitted by end users is constrained solely by the wireless network capacity provided by the base stations.

4. We temporarily ignore the energy consumption of base stations, while leaving the incorporation of base station energy consumption in the extended model.

3.1.2 Data Center

We consider that the service provider only owns one data center. Of the megawatts required to power a data center, a significant portion (typically 80%-85% [33]) is consumed by servers and cooling systems (which keep the servers running at an appropriate temperature for reliability reasons), modeled as follows.

- Cooling system: Multiple approaches (e.g., outside air economizer, chiller plant) may be available to keep servers cool by exhausting the generated heat to the outside. In this paper, we consider chiller plant as the only approach to cooling down servers, and assume that the cooling system incurs a power consumption that is linearly increasing with the number of active servers, i.e., $f(m) = \gamma \cdot m$ where $m \geq 0$ is the (normalized) number of active servers and $\gamma > 0$ is a certain constant depending on the chiller structure. Albeit simple, the linear relation has been found to be reasonably accurate through empirical measurement [6], [34]. Without causing ambiguity, we sometimes use $f(t) = f(m(t))$ to represent the cooling system power consumption at time t.
- Servers: We assume that the data center houses (homogeneous) servers, each of which has a normalized processing speed of 1 and incurs a power of 1 when active. The service provider can dynamically *size* its data center by turning on/off servers to adapt capacity provisioning to incoming workloads. For the sake of analysis, we ignore the toggling costs (e.g., wear-and-tear costs, switching on/off costs) that can be dealt with using separate techniques [2]. Without considering delay-sensitive interactive workloads, we denote by $W(t) \in [0, W]$ the number available servers for batch workloads at time t, where W is the total number of servers in the data center.
- Power supply: A modern data center is powered through multiple sources of energies such as grid power and renewable energy (in the form of solar, wind, etc.) [33]. Neglecting the energy leakage/loss during the transmission and conversion processes, we denote the available renewable energy for processing batch workloads at time t by $y(t) \in [0, y_{\max}]$. We consider that the service provider participates in a real-time electricity market to purchase grid power. Letting $\phi(t)$ be the real-time electricity price at time t, we can express the energy cost incurred by the purchase of grid power as $r(t) = r(\phi(t), e(t))$, where $e(t) \geq 0$ is the total amount of power purchased from utility companies (for processing batch jobs). Without considering the peak demand charging rate (which can be viewed as additional

penalty charges), the energy cost $r(t)$ can be written as $r(t) = r(\phi(t), e(t)) = \phi(t)e(t)$, which we shall use extensively as follows.

Next, we specify the control decisions and profit of the service provider. We denote the service provider's price for its batch services at time t by $p(t)$, which is held constant during the entire span of time t but may change across time slots. We quantify the jobs' service demand using the (normalized) number of servers: we use $b(t) \in [0, b_{\max}]$ to represent the batch service demand of all the subscribers at time t. We later explain by which means $b(t)$ is determined. Since batch jobs may be deferred to an *appropriate* time to process, a batch job queue is maintained in the data center, whose queue length is represented by $q(t)$. By letting $d(t) \in [0, d_{\max}]$ be the amount of batch jobs processed at time t, we can show that the batch job queue length evolves according to the following dynamics

$$q(t+1) = \left[q(t) - d(t)\right]^+ + b(t), \tag{1}$$

where $[\ \cdot\ ]^+ = \max[\cdot, 0]$. Supposing that the electricity cost increases linearly with the power drawn from the grid and given the control decision $p(t)$ and $d(t)$, we can express the service provider's profit at time t as[5]

$$h(t) = p(t)b(t) - \phi(t) \cdot \left[d(t) + f(d(t)) - y(t)\right]^+, \tag{2}$$

where $f(d(t))$ is the cooling energy consumption for batch workloads and $\phi(t)$ is the real-time electricity price at time t.

3.2 Users

In order to maximize its profit, the service provider needs to know the aggregate *demand function* of all the users in the market. While there are several approaches to modeling the users' demand (e.g., directly defining demand function), we use a *representative* user model: all the users served by base station i are consolidated into one representative user i, which then determines the *aggregate* service demand at each time t by solving a utility maximization problem [24]. To account for the time-varying number of users, we associate with each representative user i a time-dependent utility function $u_i(b_i(t), t)$, where $b_i(t)$ is the representative user i's batch service demand in response to the service provider's price $p(t)$ at time t. We use $S(t) = S(b_i(t))$, which is increasing in $b_i(t) \in [0, b_{i,\max}]$, to denote the required network capacity for submitting $b_i(t)$ amount of batch jobs. Thus, we can mathematically formulate the (net) utility maximization problem for each representative user i as follows

$$\max_{b_i(t) \in [0, b_{i,\max}]} \quad u_i(b_i(t), t) - p(t)b_i(t), \tag{3}$$

$$s.t. \qquad S(b_i(t)) \le C_i(t), \tag{4}$$

where $C_i(t)$ is the available network capacity of base station i. Denote the optimal solution to the above problem (3)(4) by $b_i^*(t)$, which is clearly a function of $p(t)$. Then, the total batch service demand of all the users can be expressed as $b(t) = \sum_{i=1}^{N} b_i^*(t)$, which can alternatively be expressed as $b(t) = b(p(t))$. We further assume that $b(t) = b(p_{\max}) = 0$.

5. As we normalize the power consumption of each active server by 1 and the the number of jobs in the batch queue can be represented by their requirement in terms of "machine-hours", the amount of processed batch workloads is equivalently quantified in terms of the power consumption.

3.3 Extension

In this subsection, we extend our model by incorporating: (1) profit of interactive services and base station power consumption; and (2) multiple service classes.

- Interactive services and base station power consumption: We use $a(t) \in [0, a_{\max}]$ to model the (aggregate) interactive service demand at time t and denote the profit per unit of interactive service demand by p_a. The profit p_a can be optimized using conventional profit-maximizing techniques [32], which is beyond the scope of our study and is exogenously determined as is. We model the power consumption of base stations, which takes up an increasingly significant portion of the total power in wireless systems [35], by assuming that base station i consumes a power of

$$e_{b,i}(t) = e_{s,i} + e_{d,i}(b(t) + a(t)), \tag{5}$$

where $e_{s,i}$ is constant (static) power and $e_{d,i}(b(t) + a(t))$ is the dynamic power that is increasing with the sum of batch and interactive service demands. While our model is applicable for time-varying electricity prices for base stations, we assume for the convenience of presentation that the base stations incur a fixed price ϕ_b over each time slot regardless of the locations. Combining the profit of interactive services as well as the base station energy cost, the service provider's net profit at time t can then be expressed as $h(t) = p(t)b(t) + p_a a(t) - \phi(t) \cdot \left[d(t) + f(d(t)) + a(t) + f(a(t)) - y(t)\right]^+ - \phi_b \sum_{i=1}^{N} e_{b,i}(t)$, where $y(t)$ becomes the on-site renewable energy that is available for processing all the workloads (rather than only for batch workloads).

- Multiple service classes: We now consider Quality-of-Service (QoS) differentiation (in terms of average queueing delay) for multiple service classes. Suppose that there are K classes of services available to the users, and different service classes are associated with different QoS requirements. Then, each representative user i's batch job demand becomes a vector $\mathbf{b_i}(t) = (b_{i,1}(t), b_{i,2}(t), \ldots, b_{i,K}(t))$, where $b_{i,k}(t)$ is the demand for class-k service at time t, and its utility function $u_i(\mathbf{b_i}(t), t)$ is defined in terms of $\mathbf{b_i}(t)$. As an example, we can define the utility function as $u_i(\mathbf{b_i}(t), t) = \alpha_{i,t} \log(1 + \sum_{k=1}^{K} \beta_k b_{i,k}(t))$, which β_k represents the relative importance of class-k services. The service provider charges a price of $p_k(t)$ for class-k services at time t. Thus, we can express the representative user i's net utility as follows

$$u_i(\mathbf{b_i}(t), t) - \sum_{k=1}^{K} p_k(t) b_{i,k}(t). \tag{6}$$

which is maximized subject to $\sum_{k=1}^{K} S_k(b_{i,k}(t)) \le C_i(t)$, where $S_k(b_{i,k}(t))$ is the required network capacity for demanding $b_{i,k}(t)$ class-k batch services. In the data center, each service class is associated with a separate queue, whose queue length dynamics evolves similarly as in (1). For QoS differentiation, the queue length of class-k batch jobs needs to be multiplied by a positive constant μ_k: a larger value of μ_k indicates a higher level of QoS, i.e., shorter average response time.

4 Dynamic Scheduling and Pricing

This section first presents an offline optimal formulation for maximizing the service provider's profit. Then, an online algorithm that can be implemented in practice is developed, followed by its performance analysis.

4.1 Offline Problem Formulation

Let $\bar{h}$ be the time average of the service provider's profit under a particular control policy implemented over a sufficiently large but finite time horizon with t_{end} time slots:

$$\bar{h} \triangleq \frac{1}{t_{end}} \sum_{t=0}^{t_{end}-1} h(t), \tag{7}$$

where $h(t) = p(t)b(t) - \phi(t) \cdot \left[d(t) + f(d(t)) - y(t)\right]^+$ is defined in (2). Similarly, we define $\bar{b} \triangleq \frac{1}{t_{end}} \sum_{t=0}^{t_{end}-1} b(t)$, and $\bar{d} \triangleq \frac{1}{t_{end}} \sum_{t=0}^{t_{end}-1} d(t)$. For notational simplicity, denote $\mathbf{z}(t) = (p(t), d(t))$ as the control decision made by the service provider at time t. The problem of maximizing the service provider's long-term profit can be formulated as follows:

$$\max_{\mathbf{z}(t), 0 \le t \le t_{end}-1} \quad \bar{h} = \frac{1}{t_{end}} \sum_{t=0}^{t_{end}-1} h(t) \tag{8}$$

$$\text{s.t.,} \quad \bar{b} \le \bar{d}, \tag{9}$$

$$b(t) = \sum_{i=1}^{N} b_i^*(t), \forall t, \tag{10}$$

$$b_i^*(t) \text{ solves (3)(4)}, \forall t \tag{11}$$

$$d(t) \le W(t), \forall t. \tag{12}$$

The constraint (9) specifies that the batch service demand needs to be accommodated over a large time scale (i.e., queue stability constraint). Although this constraint itself does not capture the average batch job delay or queue length constraint, we will show using the Lyapunov optimization technique that given any finite control parameter V, the batch job queue length can be deterministically bounded, which translates into a certain average queueing delay. The constraints (10)(11) state that the batch service demand is determined through the demand-pricing relation as in (3)(4). The constraint (12) is the data center capacity constraint. To solve the optimization problem (8)–(12), we need offline information (e.g., future renewable energy supplies, electricity prices, job demand) which, however, is unavailable in practice, suggesting the use of online algorithms that can be applied based on the currently available information only.

4.2 Online Algorithm

We present an online algorithm "Dyn-SP", whose performance is provably "good" compared to that of the optimal offline policy with T-slot lookahead information, based on the recently developed Lyapunov optimization [36]. The intuition of Dyn-SP is to trade the queueing delay for profit improvement by using the batch job queue length as a guidance for making scheduling and pricing decisions: batch service demand is reshaped using pricing as a lever to adapt to the data center management, and batch jobs are processed only when the queue length becomes sufficiently large and/or electricity prices are sufficiently low.

Algorithm 1 Dyn-SP Algorithm

1: At the beginning of every time slot t, observe the current environment information (i.e., $\phi_i(t)$, $W(t)$, $C_i(t)$ and $y(t)$) and the current queue length $q(t)$

2: Choose $p(t) \in [0, p_{\max}]$ to minimize

$$b(t) \cdot \left[q(t) - Vp(t)\right] = b(p(t)) \cdot \left[q(t) - Vp(t)\right], \tag{13}$$

where $b(t) = b(p(t))$ is the demand function for batch services satisfying (10)

3: Choose $d(t) \in [0, d_{\max}]$ to minimize

$$V \cdot r(\phi(t), \left[d(t) + f(d(t)) - y(t)\right]^+) - q(t)d(t) \tag{14}$$

where $r(\phi(t), \left[d(t) + f(d(t)) - y(t)\right]^+)$ is the electricity cost

4: Update $q(t)$ according to (1)

We describe Dyn-SP in Algorithm 1, which is purely online and requires only the current information and queue lengths as the input. Solving (13) is a much simpler problem than directly solving (8)–(12). Even though (13) may not be convex in the pricing decision $p(t) \in [0, p_{\max}]$ (for example, $-p(t)b(t)$ may be non-convex in $p(t)$), the computation complexity of minimizing (13) is still affordable for the service provider because: first, it only involves one decision variable; and second, minimizing (13) is performed only once every time slot (which, in practice, may correspond to one hour). The parameter $V > 0$ is a control variable which we refer to as profit-delay parameter, and it can be tuned to different values to trade the queueing delay for the service provider's long-term profit.

• Effect of V on the service provider's pricing decision: Let us consider two extreme cases: $V \to 0$ and $V \to \infty$. When $V \to 0$, the batch jobs cannot tolerate any delays (i.e., essentially they become interactive jobs) and hence, as can be seen from (13), the service provider always sets $p(t) = p_{\max}$ such that no one uses its batch service. On the other hand, when $V \to \infty$, average queueing delay is not a concern and we can notice from (13) that the service provider always chooses its price $p(t) \in [0, p_{\max}]$ such that its profit $b(p(t))p(t)$ is maximized.

• Effect of V on the service provider's scheduling decision: For simplicity, we focus on the scenario in which the electricity price is given by $\phi(t) \cdot \left[d(t) + f(d(t)) - y(t)\right]^+$, where $\phi(t)$ is the real-time electricity price and $y(t)$ is the available renewable energy supply. Then, if $\phi(t) \le \frac{q(t)}{V(1+\gamma)}$ is satisfied where γ is the required cooling power per unit number of active servers, the service provider will try to schedule as many batch jobs as possible to process. Otherwise, the service provider will wait until the electricity price is *sufficiently* low relative to the job queue length to process batch jobs. On the one hand, given a large value of V, the service provider *opportunistically* utilizes low electricity prices for its batch services, which clearly reduces the energy cost while increasing the queueing delay. On the other hand, given a small value of V, batch jobs are processed using power drawn from the electricity grid even though the electricity price is not *sufficiently low* (as can be

seen from the condition $\phi(t) \leq \frac{q(t)}{V(1+\gamma)}$). Doing so will clearly reduce the queueing delay, whereas at the same time it increases the electricity cost (and hence reduces the profit).

4.3 Performance Analysis

This subsection formally shows that, given a profit-delay parameter V, Dyn-SP is $O(1/V)$-optimal with respect to average profit compared to the optimal T-slot lookahead policy, while the queue length is bounded by $O(V)$.

4.3.1 T-Slot Lookahead Policy

As a benchmark, we present the T-slot lookahead policy, which has full knowledge of the environment information in the next (up to) T time slots. If T is sufficiently large (e.g., in the extreme case $T = t_{end}$), the T-slot lookahead policy also "approximately" (or exactly if $T = t_{end}$) maximizes the average profit in (8). We divide the time horizon of t_{end} time slots into $R \in \mathbb{Z}^+$ frames, each of which contains T time slots such that $t_{end} = RT$. In the T-slot lookahead algorithm, the service provider solves the following problem at the beginning of the r-th frame, for $r = 0, 1, \ldots, R-1$,

$$\max_{\mathbf{z}(t), rT \leq t \leq rT+T-1} \quad \frac{1}{T} \sum_{t=rT}^{t=rT+T-1} h(t) \tag{15}$$

$$\text{s.t.,} \quad \sum_{t=rT}^{t=rT+T-1} [b(t) - d(t)] \leq 0, \tag{16}$$

$$\text{Constraints (10)} - \text{(12)}. \tag{17}$$

In the problem (15)–(17), we denote the maximum of $\frac{1}{T}\sum_{t=rT}^{t=rT+T-1} h(t)$ by H_r^* and thus, the maximum profit over R frames achieved by the optimal T-slot lookahead policy is $\frac{1}{R}\sum_{r=0}^{R-1} H_r^*$.

4.3.2 Online Algorithm Analysis

Now, we present the performance analysis of Dyn-SP compared with the optimal T-slot lookahead policy. We first present the following slackness conditions.

Slackness Conditions: There exists a positive value $\delta > 0$ and a sequence of control decisions $\mathbf{z}(t) = (p(t), d(t))$ such that, for $t = 0, 1, \cdots t_{end} - 1$, the following conditions are satisfied

$$b(t) \leq d(t) - \delta, \tag{18}$$

$$d(t) \leq W(t) - \delta. \tag{19}$$

We note that the above slackness conditions are not restrictive at all. On the one hand, the condition (18) is naturally satisfied by our formulation, as the service provider can always set $p(t) = p_{\max}$ such that $b(t) = 0$. On the other hand, the condition (19) ensures that the available server resource is always enough to process all the scheduled batch jobs with a certain *slackness*. In practice, this condition is quite mild and can be easily satisfied due to data center's server over-provisioning. Next, we provide Theorem 1 to show a profit bound and queue length bound for Dyn-SP.

Theorem 1. *Suppose that the slackness conditions* (18)(19) *are satisfied for some $\delta > 0$, that the environment (e.g., $\phi(t)$, $C_i(t)$ and $y(t)$) is arbitrarily random, for $t = 0, 1, \cdots t_{end} - 1$, and that the queue length is initially zero $q(0) = 0$. Then, the following statements hold.*

a. *The queue length are bounded. For any time slot $t = 0, 1, \cdots t_{end} - 1$, we have*

$$q(t) \leq \frac{VA_3}{\delta}, \tag{20}$$

where $V \geq 0$ and A_3 is a finite number defined in the supplementary material, which is available in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TMC.2013.57.

b. *For any $T \in \mathbb{Z}^+$ and $R \in \mathbb{Z}^+$ such that $t_{end} = RT$, the profit achieved by Dyn-SP satisfies*

$$\bar{h}^* \geq \frac{1}{R}\sum_{r=0}^{R-1} H_r^* - \frac{B + D(T-1)}{V}, \tag{21}$$

where $\bar{h}^*$ is the (average) profit achieved by Dyn-SP, B and D are finite constants defined in the supplementary material, available online, and H_r^* is the maximum profit in the r-th frame achieved by the T-slot lookahead policy.

Proof. The proof is provided in the supplementary material, available online. □

Theorem 1 shows that, given a profit-delay parameter V, Dyn-SP is $O(1/V)$-optimal with respect to the average profit compared to the optimal T-slot lookahead policy, while the queue length is bounded by $O(V)$. Note that the upper bound on the profit loss $\frac{B+D(T-1)}{V}$ grows with the value of T (i.e., information window size of the *oracle* which has the future information), which can be intuitively understood as follows: with more information about the future environment, the profit achieved by the offline algorithm increases, thereby resulting in an enlarged profit gap. One may wonder if the derived performance bounds in Theorem 1 are tight. While admittedly the queue length bound and the profit bound may not be tight for all scenarios, the established bounds hold under very mild slackness conditions (18)(19) that almost all the practical scenarios can easily satisfy. Thus, our analysis is still useful and it presents a robust performance guarantee. We will also consider case studies using traces to provide more accurate estimates of profits and queueing delays.

Monotonically increasing profit-delay parameters. Our preceding analysis as well as prior research [13], [17], [29], [30] requires a constant value of V in order to show performance bounds in Theorem 1. In other words, the performance bounds hold only for a *stabilized* value of V, even though it may be time consuming to identify an appropriate value of V based on trial-and-error. This motivates us to derive the performance bounds of Dyn-SP by explicitly considering time-varying values of V that may be used in practical scenarios. For the convenience of presentation, we allow the control parameter V to be adjusted in a monotonically increasing manner every T time slots: $0 < V_0 \leq V_1 \leq \cdots \leq V_{R-1}$, where V_r is the profit-delay parameter chosen for the r-th frame (consisting of T time slots). Specifically, given monotonically increasing values of V, we can prove that the queue length bound is $q(t) \leq \frac{V_{R-1}A_3}{\delta}$, where V_{R-1} is the maximum profit-delay parameter and A_3 is a finite number defined in the supplementary material, available online, while the profit bound

becomes $\bar{h}^* \geq \frac{1}{R}\sum_{r=0}^{R-1} H_r^* - \frac{B+D(T-1)}{R}\sum_{r=0}^{R-1} V_r$. The proof is omitted due to space limitations. While considering arbitrary selections of V over the time is left as our future work, we note that such choices of $0 < V_0 \leq V_1 \leq \cdots \leq V_{R-1}$ are useful when the profit gets more critical while the queueing delay becomes less stringent (e.g., initially, the queueing delay is too small and can be relaxed to some extent without affecting user experiences).

5 Performance Evaluation

We perform a simulation study to evaluate Dyn-SP using synthetic data as well as real-world traces of hourly electricity prices and renewable energy supplies. We conduct the following sets of experiments:

- Impact of V: We show the impact of the profit-delay parameter V on profit maximization and delay performance.
- Demand reshaping: We show that pricing proactively reshapes the batch service demand in according to the job queue length in the data center.
- Algorithm comparison: We compare Dyn-SP with three other algorithms (i.e., StatPricing, ServerOnly, BestEffort) explained later.
- Extension: We extend the results to incorporate interactive services and multiple service classes.

The simulation results show that: (1) Dyn-SP maximizes profit while bounding the queue length; (2) Compared to the existing algorithms, Dyn-SP can significantly increase the service provider's profit while achieving (almost) the same average delay performance; and (3) Dyn-SP can be effectively extended to incorporate interactive services as well as multiple service classes for QoS differentiation.

5.1 Setup

In this subsection, we introduce *default* settings that are used throughout the simulations unless otherwise stated.

- Data center: We re-scale real-world traces of hourly electricity prices and renewable energy supplies in California, USA [37]. The total number of servers in the data center is normalized to 10. Based on empirical data provided in [33], we assume that the cooling system consumes a power equal to 0.75 times the server power, i.e., $f(m) = 0.75 \cdot m$, where m is the number of active servers.
- User: For simplicity, we assume that there are 10 base stations (each of which may correspond to multiple physical base stations in practice). The utility function for the representative user i is $u_i(b_i(t), t) = \alpha_{i,t}\log(1 + b_i(t))$, in which the demand state $\alpha_{i,t}$ is assumed to be independently and uniformly distributed in $[0, 1]$. The normalized number of available servers for processing batch workloads is uniformly distributed in $[6, 10]$. We assume that a unit of network capacity is required for submitting a unit of batch service demand to the service provider, and that the available network capacity $C_i(t)$ for batch services follows a uniform distribution in $[0, 10]$.

Despite the lack of access to real-world data traces, we note that the above setting captures a time-varying environment and is sufficient for evaluating the benefits of Dyn-SP.

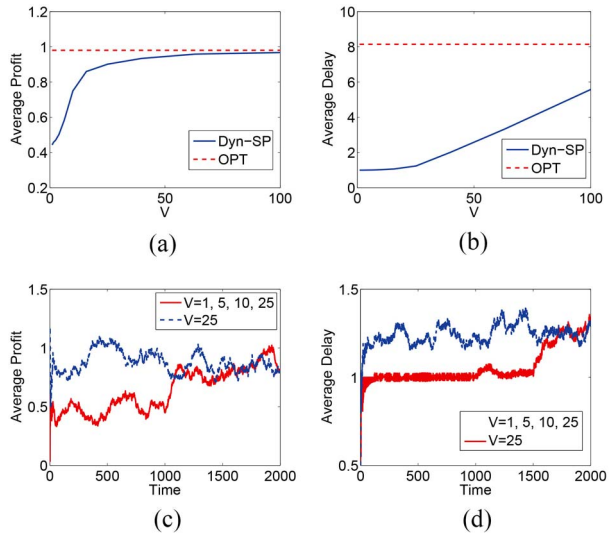

Fig. 2. Impact of V. (a) Profit versus V. (b) Delay versus V. (c) Profit versus time. (d) Delay versus time.

5.2 Simulation Results

In this subsection, we provide detailed simulation results.[6]

5.2.1 Impact of V

We first compare Dyn-SP with the optimal offline algorithm (denoted by OPT) that has complete offline information over the $t_{end} = 2000$ time slots, in terms of the average profit and average delay. Note that, as can be seen from the constraint (9), OPT is optimal in the sense of maximizing the profit, while it does not take into consideration the delay performance during the optimization process. Fig. 2 shows that when V exceeds approximately 25, the average profit is already approximately 90% of the maximum profit achieved by the optimal offline algorithm (while the average delay is still acceptable, i.e., most batch jobs are processed without significant delays).[7] It can also be seen that when V increases, the average profit increases at the expense of increasing the average queueing delay, which verifies Theorem 1. Now, we show in Fig. 2(c) and (d) the impact of monotonically increasing V over the course of operation. Specifically, we change V every 500 time slots (i.e., hours) and present the moving average profit and delay (averaged over the past 120 time slots). We observe from Fig. 2(c) and (d) that, by choosing a small V initially, the average profit is quite small whereas it can be significantly increased later by increasing the value of V (at the expense of increasing the average delay). This corroborates the flexibility of dynamically tuning V to adjust the tradeoff between profit and delay. In Fig. 2, we do not show the optimal offline algorithm with T-step lookahead information, because it cannot possibly achieve a profit greater than the optimal offline algorithm with complete information, compared to which Dyn-SPalready achieves a close-to-maximum profit.

6. Unless otherwise stated, the average values at time $t = 1, 2, \cdots$ are obtained by summing up all the values up to time t and then dividing the sum by t.

7. Due to the discrete-time model, the minimum delay is *one* time slot in our study.

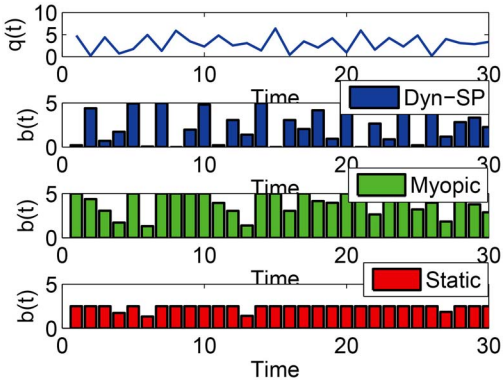

Fig. 3. Queue length and demand history ($V = 10$).

5.2.2 Demand Reshaping

Let us illustrate how Dyn-SP *reshapes* the service demand to adapt it to data center management. Fig. 3 shows a 30-hour snapshot of queue lengths and batch service demands under three different pricing schemes, i.e., Dyn-SP, Myopic (which maximizes the instantaneous profit without considering the impact of the pricing decision on the queue dynamics), and Static (which uses a fixed price of $p(t) = 0.4$, for $t = 0, 1, 2, \cdots 1999$). The demand state $\alpha_{i,t}$ is set as a constant 0.5 to isolate the demand state randomness and to highlight the role of queue length in setting the price. We observe from Fig. 3 that Dyn-SP can adapt the demand to the queue length: in general, the service provider will set a lower price to attract a higher batch service demand when the queue length is smaller, whereas a higher price will be used to suppress the batch service demand when the queue length is larger (such that excessive delay can be avoided). By contrast, Myopic and Static pricing schemes ignores the queue length information and thus, the batch service demands under these two pricing schemes are only constrained by the available network capacity.

5.2.3 Algorithm Comparison

We next compare Dyn-SP against three other algorithms described as follows.

- StatPricing: StatPricing uses a fixed price at all times, while it uses Step 3 in Algorithm 1 to make scheduling decisions.
- ServerOnly: ServerOnly ignores the cooling system energy consumption and only considers server energy consumption when making scheduling decisions.
- BestEffort: BestEffort tries to maximize the service provider's instantaneous profit and processes the submitted batch jobs as soon as possible.

We show in Fig. 4 the profit comparison subject to (almost) the same average delay constraint of 1.15. It shows that Dyn-SP outperforms all the other algorithms in terms of the average profit. In particular, Fig. 4(a) shows the importance of dynamic pricing, which reshapes the batch service demand to the data center operation, while Fig. 4(b) indicates that an integrated approach to managing the data center, i.e., considering both cooling system and server energy consumption, is important for profit maximization. Moreover, as shown in Fig. 4(c), the BestEffort algorithm is inferior to Dyn-SP in terms of profit maximization, since it neglects the queue length information and hence it cannot reshape the batch job service demand to data center operation or exploit low electricity prices. To show the robustness of Dyn-SP against the random environment, we also consider synthetic data by assuming uniformly distributed electricity prices and available renewable energy supplies. The results are similar and hence omitted for brevity.

5.2.4 Extension

We perform two extended simulations by incorporating: (1) profit of interactive services and base station power consumption; and (2) multiple service classes.

We assume that the average normalized profit for processing each unit of interactive workload is 0.15, and that the interactive service demand is uniformly distributed in $[0, 3]$. The normalized static power of each base station 0.1, the dynamic power increases linearly with the total service demand (i.e., batch plus interactive), and the electricity price for base stations is fixed as 1. The other settings are used by default. Fig. 5 shows the comparison between Dyn-SP and the optimal offline algorithm with complete offline information over the entire horizon of $t_{end} = 2000$ time slots. Like in Fig. 2, it can be seen that when V exceeds around 50, the average profit is already approximately 90% of the maximum profit achieved by the optimal offline algorithm (while the average delay is still satisfactory). By comparing Figs. 2 and 5, we notice that the total profit considering the interactive services and base station power consumption is even lower than that of only considering batch services. This is because the base station consumes a significant amount of power and incurs a large operational cost. We also compare Dyn-SP against the other algorithms and, due to the similarity with Fig. 4, the results are not shown for brevity.

The proposed Dyn-SP can be extended to multiple service classes and provide differentiated QoS in terms of the average delay. For the ease of illustration, we consider only two service classes, and assume that the utility function of representative user i is $u_i(\mathbf{b_i}(t), t) = \alpha_{i,t} \log(1 + b_{i,1}(t) + 2b_{i,2}(t))$. Without loss of generality, we assume that class-2 service requires a lower queueing delay on average than class-1 service. Intuitively, the service provider sets a higher price for class-2 service than class-1 service. This can be seen from Fig. 6, which also shows that the average delay for class-2 service is lower than that for class-1 service. Because of space limitations, we omit the results showing the tradeoff between the average profit and delay.

6 Conclusion

In this paper, we considered a profit-maximizing wireless service provider "selling" cloud computing services to its subscribers. Focusing on batch services, we proposed a provably-efficient online scheduling and pricing algorithm, Dyn-SP, which can be implemented based on the currently available information. We proved that, compared to the optimal offline algorithm with future information, Dyn-SP produces a close-to-optimal long-term profit while bounding the job queue length in the data center. A simulation study was performed to demonstrate the effectiveness of Dyn-SP. In particular, it was shown

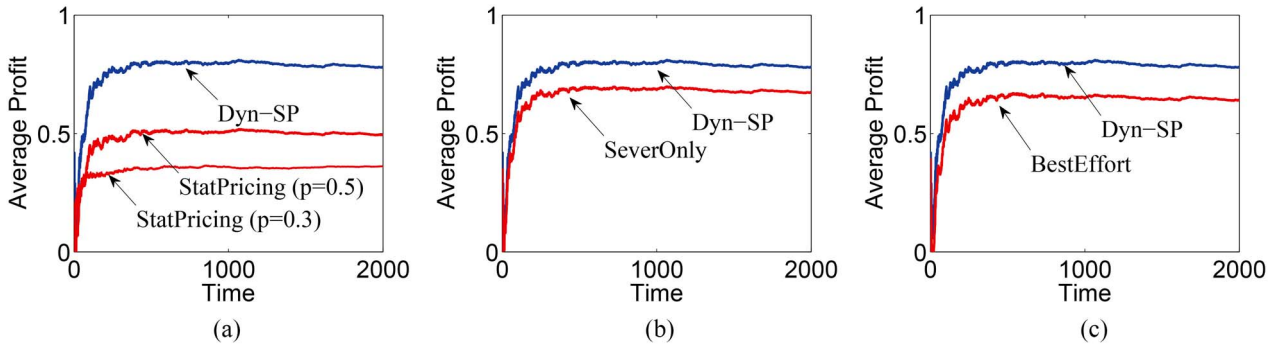

Fig. 4. Algorithm comparison. (a) Compare Dyn-SP with StatPricing. (b) Compare Dyn-SP with ServerOnly. (c) Compare Dyn-SP with BestEffort.

both analytically and numerically that a desired tradeoff between the profit and queueing delay can be obtained by appropriately tuning the control parameter. Our results also indicated that, compared to the other algorithms which neglect demand-side management, cooling system energy consumption, or the queue length information, Dyn-SP achieves a higher average profit while incurring (almost) the same average delay.

ACKNOWLEDGMENTS

This work was partially supported by the US National Science Foundation under Grants 0830556 and 1218136.

REFERENCES

[1] *AT&T Cloud Services* [Online]. Available: http://www.synaptic.att.com
[2] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," in *Proc. IEEE INFOCOM*, Shanghai, China, 2011.

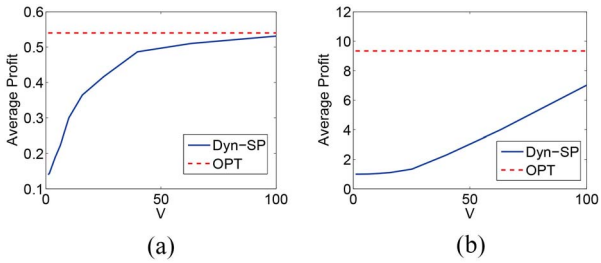

Fig. 5. Impact of V (considering profit of interactive services and base station power consumption). (a) Average profit. (b) Average delay.

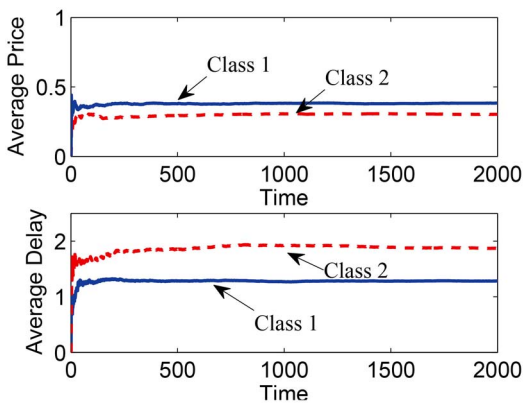

Fig. 6. Average price and delay for two service classes.

[3] L. Chen, N. Li, and S. H. Low, "On the interaction between load balancing and speed scaling," in *Proc. ITA Workshop*, 2011.
[4] R. Ferzli and I. Khalife, "Mobile cloud computing educational tool for image/video processing algorithms," in *Proc. IEEE DSP/SPE*, Sedona, AZ, USA, 2011.
[5] *BMC Workload Automation: Helping Cloud Computing take Flight* [Online]. Available: http://documents.bmc.com/products/documents/62/56/286256/286256.pdf
[6] Z. Liu *et al.*, "Renewable and cooling aware workload management for sustainable data centers," in *Proc. SIGMETRICS*, London, U.K., 2012.
[7] Y. Wen, W. Zhang, and H. Luo, "Energy-optimal mobile application execution: Taming resource-poor mobile devices with cloud clones," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, 2012.
[8] C. Maglaras and A. Zeevi, "Pricing and capacity sizing for systems with shared resources: Approximate solutions and scaling relations," *Manage. Sci.*, vol. 49, no. 8, pp. 1018–1038, Aug. 2003.
[9] H. Mohsenian-Rad, V. W. S. Wong, J. Jatskevich, R. Schober, and A. Leon-Garcia, "Autonomous demand side management based on game-theoretic energy consumption scheduling for the future smart grid," *IEEE Trans. Smart Grid*, vol. 1, no. 3, pp. 320–331, Dec. 2010.
[10] C. Joe-Wong, S. Sen, S. Ha, and M. Chiang, "Optimized day-ahead pricing for smart grids with device-specific scheduling flexibility," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 6, pp. 1075–1085, Jun. 2012.
[11] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, "Greening geographical load balancing," in *Proc. SIGMETRICS*, San Jose, CA, USA, 2011.
[12] M. Lin, Z. Liu, A. Wierman, and L. L. H. Andrew, "Online algorithms for geographical load balancing," in *Proc. IGCC*, 2012.
[13] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan & Claypool, 2010.
[14] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, 2012.
[15] D. Huang, P. Wang, and D. Niyato, "A dynamic offloading algorithm for mobile computing," *IEEE Trans. Wireless Commun.*, vol. 11, no. 6, pp. 1991–1995, Jun. 2012.
[16] N. U. Prabhu, *Foundations of Queueing Theory*. Boston, MA, USA: Kluwer Academic, 1997.
[17] S. Ren, Y. He, and F. Xu, "Provably-efficient job scheduling for energy and fairness in geographically distributed data centers," in *Proc. ICDCS*, Macau, China, 2012.
[18] B. Guenter, N. Jain, and C. Williams, "Managing cost, performance and reliability tradeoffs for energy-aware server provisioning," in *Proc. IEEE INFOCOM*, 2011.
[19] L. Rao, X. Liu, L. Xie, and W. Liu, "Reducing electricity cost: Optimization of distributed internet data centers in a multi-electricity-market environment," in *Proc. IEEE INFOCOM*, 2010.
[20] N. Buchbinder, N. Jain, and I. Menache, "Online job migration for reducing the electricity bill in the cloud," in *Proc. IFIP Netw.*, 2011.
[21] Y. Xue, B. Li, and K. Nahrstedt, "Optimal resource allocation in wireless ad hoc networks: A price-based approach," *IEEE Trans. Mobile Comput.*, vol. 5, no. 4, pp. 347–364, Apr. 2006.

[22] B. Wang, Z. Han, and K. J. R. Liu, "Distributed relay selection and power control for multiuser cooperative communication networks using Stackelberg game," *IEEE Trans. Mobile Comput.*, vol. 8, no. 7, pp. 975–990, Jul. 2009.
[23] I. C. Paschalidis and J. N. Tsitsiklis, "Congestion-dependent pricing of network services," *IEEE/ACM Trans. Netw.*, vol. 8, no. 2, pp. 171–184, Apr. 2000.
[24] P. Hande, M. Chiang, R. Calderbank, and S. Rangan, "Network pricing and rate allocation with content provider participation," in *Proc. IEEE INFOCOM*, 2011.
[25] S. Sen, Y. Jin, R. Guérin, and K. Hosanagar, "Modeling the dynamics of network technology adoption and the role of converters," *IEEE/ACM Trans. Netw.*, vol. 18, no. 6, pp. 1793–1805, Dec. 2010.
[26] I. Menache, A. Ozdaglar, and N. Shimkin, "Socially optimal pricing of cloud computing resources," in *Proc. VALUETOOLS*, Brussels, Belgium, 2011.
[27] Q. Wang, K. Ren, and X. Meng, "When cloud meets ebay: Towards effective pricing for cloud computing," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, 2012.
[28] S. Ha, S. Sen, C. Joe-Wong, Y. Im, and M. Chiang, "Tube: Time-dependent pricing for mobile data," in *Proc. SIGCOMM*, Helsinki, Finland, 2012.
[29] Y. Guo, Z. Ding, Y. Fang, and D. Wu, "Cutting down electricity cost in internet data centers by using energy storage," in *Proc. IEEE GLOBECOM*, Houston, TX, USA, 2011.
[30] R. Urgaonkar, B. Urgaonkar, M. J. Neely, and A. Sivasubramaniam, "Optimal power cost management using stored energy in data centers," in *Proc. SIGMETRICS*, San Jose, CA, USA, 2011.
[31] C. Ren, D. Wang, B. Urgaonkar, and A. Sivasubramaniam, "Carbon-aware energy capacity planning for datacenters," in *Proc. MASCOTS*, Washington, DC, USA, 2012.
[32] H. R. Varian, *Microeconomic Analysis*. New York, NY, USA, W. W. Norton & Company, 1992.
[33] K. Kant, "Data center evolution: A tutorial on state of the art, issues, and challenges," *Comput. Netw.*, vol. 53, no. 17, pp. 2939–2965, 2009.
[34] T. Mukherjee, A. Banerjee, G. Varsamopoulos, S. K. S. Gupta, and S. Rungta, "Spatio-temporal thermal-aware job scheduling to minimize energy consumption in virtualized heterogeneous data centers," *Comput. Netw.*, vol. 53, no. 17, pp. 2888–2904, Dec. 2009.
[35] J. Lorincz, T. Garma, and G. Petrovic, "Measurements and modelling of base station power consumption under real traffic loads," *Sensors*, vol. 12, no. 4, pp. 4281–4310, 2012.
[36] M. J. Neely, *Universal Scheduling for Networks with Arbitrary Traffic, Channels, and Mobility* [Online]. Available: http://arxiv.org/abs/1001.0960
[37] *California ISO* [Online]. Available: http://www.caiso.com/

Shaolei Ren received his B.E., M.Phil., and Ph.D. degrees, all in electrical engineering, from Tsinghua University in July 2006, Hong Kong University of Science and Technology in August 2008, and the University of California, Los Angeles, in June 2012, respectively. Since August 2012, he has been with the School of Computing and Information Sciences, Florida International University, as an Assistant Professor. His current research interests include optimization of computer systems and network economics. He received the Best Paper Award from IEEE International Conference on Communications in 2009.

Mihaela van der Schaar is Chancellor's Professor in the Electrical Engineering Department at the University of California, Los Angeles. Her current research interests include multimedia systems, networking, communication, and processing, dynamic multi-user networks and system designs, online learning, network economics, and game theory. She is a Distinguished Lecturer of the Communications Society for 2011-2012, the Editor in Chief of the *IEEE Transactions on Multimedia* and a member of the Editorial Board of the *IEEE Journal on Selected Topics in Signal Processing*. She received a US NSF CAREER Award (2004), the Best Paper Award from the *IEEE Transactions on Circuits and Systems for Video Technology* (2005), the Okawa Foundation Award (2006), the IBM Faculty Award (2005, 2007, 2008), and the Most Cited Paper Award from *EURASIP: Image Communications Journal* (2006). She was formerly an Associate Editor for the *IEEE Transactions on Multimedia, Signal Processing Letters, Circuits and Systems for Video Technology, Signal Processing Magazine* etc. She received three ISO awards for her contributions to the MPEG video compression and streaming international standardization activities, and holds 33 granted U.S. patents.