Structure-Aware Stochastic Storage Management in Smart Grids

Yu Zhang, Mihaela van der Schaar

***Abstract*—Demand-side management has been proposed as an important solution for improving the energy consumption efficiency in smart grids. However, traditional pricing-based demand-side management methods usually rely on the assumption that the statistics of the system dynamics (e.g. the time-varying electricity price, the arrival distribution of consumers' demanded load) are known a priori, which does not hold in practice. In this paper, we propose a novel price-aware energy storage management algorithm for consumption scheduling which, unlike previous works, can operate optimally in systems where such statistical knowledge is unknown. We consider a power grid system where each consumer is equipped with an electrical energy storage device. Each consumer proactively determines how much energy to purchase from the energy producers by taking into consideration the time-varying and a priori unknown system dynamics, in order to maximize its own energy consumption utility. We first formulate the real-time energy storage management and demand response of the consumers as a Markov decision process and then propose an online learning algorithm that enables the consumers to learn the unknown system dynamics on-the-fly and have their energy storage management policies converge to the optimum. Our simulation results validate the efficacy of our algorithm, which helps consumers achieve higher average utility as opposed to other state-of-the-art online learning algorithms and energy storage management algorithms.**

***Index Terms*—Markov Decision Process, Post-decision State, Smart Grid, Load Scheduling, Energy Storage.**

I. Introduction

With the rapid progress of information and communication technologies, such as advanced metering, bi-directional communication, distributed power generation and storage, etc., energy-consumption scheduling (ECS), a demand-side management (DSM) technique, is becoming a key smart grid solution that enables efficient use of electric energy [1], improves the stability of the power system, and lowers energy production costs in the long run [2].

There exists a large body of literature on ECS, see e.g. [1]-[13]. Depending on the operating entity who performs the management, existing ECS methods can be classified into two categories. The first category of ECS is based on direct load control (DLC) [3],[4]: the utility companies install switches or thermostats on top of the existing metering infrastructure, which enable them to (directly) modify the operations of appliances during peak hours. For instance, [3] studies optimal centralized energy reallocation in smart grids; [4] investigates the coordination of charging plug-in hybrid electric vehicles with other electric appliances. Although utility-based DLC methods have been effective in smoothing peak load, they incur frequent interruptions to the normal use of the household appliances, as the control only tries to smooth the real-time load but neglects whether it fulfills the consumer's demand. For instance, when warranted by capacity shortage during the summer, a consumer's central air conditioning system will be turned down or cycled by the utility company, while the exact days and the length of the cycling period are not known and controlled by the consumer [8]. More importantly, when providing the centralized control, DLC methods usually shield the consumer heterogeneity and prevent consumers from making price-aware decisions in order to effectively (and more flexibly) perform individual consumption scheduling based on their personal demands. This further reduces the efficiency of the smart grid.

Due to the aforementioned problems of DLC, the second category of ECS, which is based on dynamic pricing, has gained increased popularity in recent years [1],[2],[5]-[13]. The basic principle of dynamic pricing is to adaptively adjust the retail price of electric energy according to the real-time variation of the production capacity and the load demands. Although not directly controlling the energy usage pattern of individual consumers, appropriate pricing can provide them effective economic incentives to schedule their consumptions more efficiently and thus help the energy producer to better procure electric energy [2].

The dynamic pricing literature can be sub-divided into two directions. The first direction takes the utility companies' perspective and designs effective pricing strategy in order to maximize the social welfare, i.e. the sum benefit of all consumers in the smart grid or the revenue of the utility company (or energy producer) from selling electric energy [6]-[8]. The second direction, which is our focus in this paper, designs effective price-aware consumption/demand scheduling policies for individual or groups of consumers, which maximize their individual benefit, given the exogenously determined (and fixed) pricing strategy from the utility company, e.g. [9]-[14].

Most existing price-aware consumption scheduling algorithms have put their focus on myopically maximization of consumers' immediate benefit, based on the instantaneous electricity price and load demand, e.g. [9]-[11]. However in the smart grid, the consumption scheduling decisions have strong temporal correlations. That is, a consumer's current consumption scheduling decision will not only affect its immediate benefit, but also its load demand and benefit in the future [2]. Neglecting such temporal correlation prevents the myopic

Yu Zhang is with Online Service Division, Microsoft, Sunnyvale, CA 94085, USA ,e-mail: yuzhan@microsoft.com.

Mihaela van der Schaar is with the Department of Electrical Engineering, University of California, Los Angeles, CA 90095, USA, e-mail: mihaela@ee.ucla.edu.

TABLE I
COMPARISON OF THE EXISTING LITERATURE AND OUR WORK

	[3],[4]	[6]-[8]	[9],[10]	[1],[2],[12],[13]	[19]	Our work
ECS approach	DLC	Dynamic pricing	Dynamic pricing	Dynamic pricing	Dynamic pricing	Dynamic pricing
Optimizing entity	Utility company	Utility company	Individual consumer	Individual consumer	Utility company	Individual consumer
Optimization criterion	Myopic; Foresighted	Myopic; Foresighted	Myopic	Foresighted	Foresighted	Foresighted
Information of system dynamics	Known	Known	Known	Known	Unknown	Unknown
Online learning	No	No	No	No	Yes	Yes
Considered dynamics	Load demand	Load demand	Load demand	Load demand	Load demand	Load demand; energy price; environment dynamics

optimization from performing well in the long run.

There are only a few works which use foresighted optimization to design the consumption scheduling algorithms [1],[2],[12]-[15]. Most of them use electricity price predictions and assume that the statistical knowledge of the underlying system dynamics (e.g. the temporal variation of the electricity prices, the arrival distribution of the consumer's load demand) is known. However, practical smart grid systems face many (environmental) unknowns, such as changing weather, diverse reactions of consumers to real-time prices, intermittent renewable energy sources (e.g. small wind farms, household with solar panels, etc.), for which the statistical knowledge cannot be reliably obtained by consumers a priori. Therefore, the efficacy of the methods proposed in these works, which rely on specific models of the system dynamics, may result in poor performance (i.e. low benefit received by individual consumers) in practice.

In this paper, we propose foresighted price-aware consumption scheduling algorithms based on energy storage management, which can operate optimally in time-varying and unknown environments. In particular, we consider a power grid where each consumer is equipped with an energy storage device [20]. The consumers purchase electric energy from an electricity market where the electricity price varies over time. Hence, each individual consumer performs consumption scheduling by proactively determining how much electric energy to purchase at each moment of time, given its real-time load demand, its current energy storage, and the electricity price. By rigorously formulating the consumer's decision problem as a Markov Decision Process (MDP), we then propose an efficient online learning algorithm that enables each consumer to learn the optimal storage management policy for consumption scheduling, which maximizes its personal benefit in the long run.

The differences between our work and the existing literature on ECS are summarized in Table I. The main contributions of our work lie in the following aspects. First, we assume that both the electricity price and the arrival of consumers' demands vary dynamically over time, and importantly the statistical knowledge of these (Markovian) dynamics is not known a priori. In order to cope with such unknown time-varying system dynamics, we propose, in our online learning algorithm, a decomposition of the (offline) value iteration and (online) reinforcement learning based on factoring the system dynamics into an a priori known and an a priori unknown component. This is achieved by generalizing the concept of a post-decision state (PDS) [28], which is an intermediate state that occurs after the known dynamics take place but before the unknown dynamics take place. A key advantage of the proposed PDS learning method is that it exploits partial information about the smart grid system and the structure of the storage management problem and thus, less information needs to be learned than when using conventional reinforcement learning algorithms such as Q-learning, actor-critic etc. [25]. Importantly, under certain conditions, it obviates the need for action exploration, which significantly improves the adaptation speed and the runtime performance as compared to conventional reinforcement learning algorithms which lose significant performance during the (very long) exploration state.

The remainder of this paper is organized as follows. In Section II, a rigorous MDP framework is proposed to formulate the storage management problem in the smart grid. In Section III, we describe a novel PDS online learning algorithm that optimally solves the storage management problem and study the convergence property of the learning algorithm. Section IV extends the proposed algorithm to solve the storage management problem in an alternative scenario where the demand loads of individual consumers are deferrable. Section V shows the numeric results. Section VI discusses various extensions of our proposed framework. We conclude in Section VII.

II. SYSTEM MODEL

A. System Setting

This section describes the smart grid system considered in this paper.

We consider an infinite-horizon discrete time model, where time slots are indexed by $t = 0, 1, \ldots$. For instance in [1], the length of each time slot is assumed to be one hour, in which the sale price of electricity changes once. Similar to [10],[13], we consider an electricity market where distributed power grids purchase electric energy from distributed energy producers. A market consists of three different types of entities: a market operator, producers and consumers:

(1) The *producers* represent the distributed entities who generate and sell energy in this wholesale market to the power grids [10],[17]. Examples of such producers include small wind farms, households with solar panels, power plants, etc.

(2) The *consumers* are the end users owning a number of residential appliances, e.g. electrical vehicles, air conditions, dishwashers, etc. Each consumer purchases electric energy from the producers on the market for its own consumption. There are several assumptions imposed on consumers in this paper.

- The continuum model is used to capture the large consumer population [1]. The continuum model approximates well the real user population if there is a sufficiently large number of consumers in the market. Hence, each consumer is infinitesimally small and its storage management policy does not interfere with the decisions of other consumers.
- Since each consumer is infinitesimally small, its storage management policy does not influence the electricity price in the market as well.
- The consumer's load demands from different time periods are not correlated with each other.

(3) The *market operator* is an independent and non-profit entity who operates the market. It manages the physical infrastructure and the electricity trading in the market. It also determines the trading price of the electricity. Note that the market operator does not directly engage in and make profits from any electricity transaction among the producers and consumers. It only operates the trading platform of the market to make the transactions feasible.

In this paper, we specifically focus on designing optimal storage management policies for strategic consumers, while assuming that the other entities in the market (i.e. the producers and the market operator) are following given strategies.[1] Since we assumed a continuum consumer model and thus the storage management policies of different consumers do not interfere with the decisions of each other, we focus on the analysis of one representative consumer in the rest of this paper. This is an approach commonly adopted in the literature analyzing continuum user populations, e.g. [17].

Before proposing the formal model, we first summary the interaction of the consumer with other entities in the market at each time slot t:

(1) At the beginning of the time slot, the market operator publishes the unit electricity price in the current time slot. The unit price is denoted as $q^{(t)} \in \mathcal{Q}$, where $\mathcal{Q}$ is a finite set of possible electricity prices [17].[2]

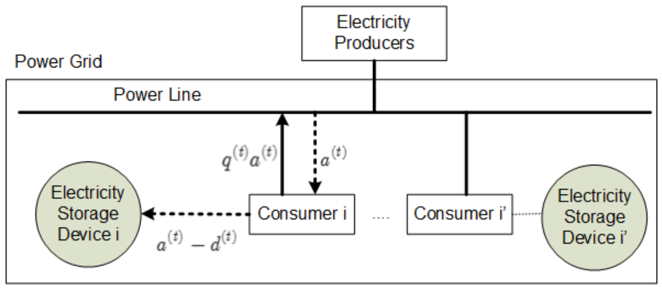


Fig. 1. The electricity market

(2) The consumer's total demanded load (from all its appliance) in the current time slot is denoted as $d^{(t)} \in [0, D] \triangleq \mathcal{D}$, where D is a constant upper limit. The load is assumed to be non-deferrable in our analysis, i.e. the current demand cannot be postponed and get fulfilled in future time slots [26]. Also, it should be noted that the actual amount of demand from all the consumers can exceed the production capacity of the smart grid system. Therefore the part of demand that is not able to be fulfilled by the smart grid system will be dropped.[3][4]

(3) The consumer purchases an amount $a^{(t)} \in [0, A] \triangleq \mathcal{A}$ of electric energy from the electricity producers to fulfill its demand coming in this time slot, where A represents the largest allowable purchasing amount [20]. In practice, the energy usage is measured to an accuracy of 0.01mW [21], and hence $\mathcal{A}$ represents a finite set. When a consumer makes the electricity purchase, which happens at the beginning of the time slot, we assume that it does not know the exact value of its demand that comes in the current time slot. This puts our work into a sharp contrast against the existing literature, where the exact demand needs to be specified and known by the consumer in order to determine its purchasing amount.

A significant problem embedded in such electricity market is that the production capacity of the distributed electricity producers varies drastically over time and is often highly unpredictable compared to the large power plants because they rely on intermittent resources like wind and sunshine [9], which in turn introduce significant fluctuations on the unit electricity price $q^{(t)}$. The stability of the power grids thus depends on having balanced electricity supply and demand at any given time. In order to achieve this balance, the consumer schedules its demand through the help of electricity storage devices (e.g. batteries, plug-in hybrid electric vehicle, etc.). The basic principle of the consumer's storage management in each time slot t is described as follows:

- If the consumer purchases more electricity than its demand, i.e. $a^{(t)} > d^{(t)}$, it stores the surplus $a^{(t)} - d^{(t)}$ in

[1]It should be noted that the stochastic control and online learning solutions proposed in this paper can be easily extended to the design and analysis of the strategic operations of entities other than the consumers. We relegate such extension as future works.

[2]In this paper, we focus on the strategic energy purchase of the consumers and assume that the energy price is determined exogenously. That is, the market operator follows some static policy to set up the unit electricity price in each time slot. The analysis and design on the adaptive pricing policy of the market operator remains as an interesting topic and is relegated to our future research.

[3]The un-fulfilled load could also be redirected to larger energy providers, e.g. Pacific Gas & Electric Company, and get supplied from there.

[4]However, we would like to note that in Section IV, we do consider the scenario where the unsatisfied load in each time slot can be deferred to be fulfilled in future.

its storage device.

- If the consumer purchases less electricity than its demand, i.e. $a^{(t)} < d^{(t)}$, it covers the deficit $d^{(t)} - a^{(t)}$ using its stored electricity, whose amount is denoted as $b^{(t)}$. Since the capacity of the storage device is limited, we assume that $b^{(t)} \in [0, B] \triangleq \mathcal{B}$, where B is a constant upper bound.

A schematic representation of the considered electricity market is illustrated in Figure 1. By strategically determining its purchase $a^{(t)}$, the consumer can flexibly utilize its electricity storage to balance the electricity supply and its demand across time in order to minimizing the negative effect introduced by the fluctuation of the electricity price. In this way, its benefit from electricity consumption can be maximized.

B. Stochastic Control Formulation

In this section, we formulate the strategic storage management of the consumer as a stochastic control problem, in order to maximize its utility on the energy consumption from the smart grid system. Specifically, we define the consumer's action in each time slot t as its purchase $a^{(t)}$ and the state as a tuple $s^{(t)} \triangleq (q^{(t)}, b^{(t)}, e^{(t)})$. Here the variable $e^{(t)}$ represents the environment dynamics (e.g. time, weather) that the consumer experiences in time slot t, other than $q^{(t)}$, $d^{(t)}$, and $b^{(t)}$. Valuable information is embedded in such environment dynamics, which influences the fluctuation of the energy price and the consumer's demand over time. For instance, it is often the case that a consumer's demand in a specific time slot is influenced by the hour that this time slot is located within a day: the demand will be high during the peak hours (e.g. 6pm-12am) and be low during the off-peak hours (e.g. 12pm-6pm) [10],[18],[23]. This information can be helpful for consumers predicting how their own demand and the electricity price will change in the near future and thus used to instruct how much electricity it should purchase in the current time slot.

To make the stochastic control problem tractable, we assume that $e^{(t)}$ also takes value from a finite set, denoted by $\mathcal{E}$, and evolves as a finite-state Markov chain with the transition probability $\{p_e(e^{(t+1)} \mid e^{(t)})\}$. Similar assumptions have been taken in existing literature, e.g. [1],[17]. We also assume, as in [9], that the evolution of the electricity price follows a stationary finite-state Markov chain given the environment dynamics, with the transition probability $\{p_q(q^{(t+1)} \mid q^{(t)}, e^{(t)})\}$.[5] To capture the influence of the environment dynamics over the consumers' demand, we model $d^{(t)}$ as an i.i.d. random variable given $e^{(t)}$, with the probability distribution $\{p_d(d^{(t)} \mid e^{(t)})\}$. It is important to note that all the probability distributions are unknown a priori to the consumer and needs to be learned dynamically over time.

Given the Markovian evolution of $q^{(t)}$ and $e^{(t)}$, the stochastic control problem can be casted into a Markov Decision Process (MDP). Next, we derive the state transition probability

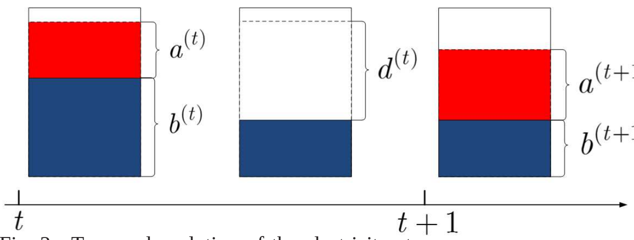


Fig. 2. Temporal evolution of the electricity storage

and value function of the MDP. The storage dynamic across time slots in this MDP is illustrated in Figure 2 and captured by the following expression:

$$b^{(t+1)} = \min\{\max\{b^{(t)} + a^{(t)} - d^{(t)}, 0\}, B\}. \tag{1}$$

The system then evolves into the next time slot $t+1$ with the new state $s^{(t+1)}$. The transition probability from $s^{(t)}$ to $s^{(t+1)}$, given the action $a^{(t)}$, is expressed as follows:

$$\begin{aligned} &p(s^{(t+1)}|s^{(t)}, a^{(t)}) \\ &= p_q(q^{(t+1)}|q^{(t)}, e^{(t)}) p_e(e^{(t+1)}|e^{(t)}) p_d(d^{(t)}|e^{(t)}) \\ &I\left(b^{(t+1)} = \min\{\max\{b^{(t)} + a^{(t)} - d^{(t)}, 0\}, B\}\right), \end{aligned} \tag{2}$$

where $I(\cdot)$ is the indicator function.

The consumer's benefit in each time slot t is determined by the amount of its electricity consumption in this time slot, denoted as $r^{(t)} \triangleq \min\{b^{(t)} + a^{(t)}, d^{(t)}\}$, i.e. the consumption cannot exceed the total amount of the stored and purchased electricity. Given this, the consumer's one-slot benefit is $f(r^{(t)})$, with $f(\cdot)$ being a function that satisfies the following conditions:

Assumption: $f(x)$ is non-decreasing and concave on x, with $\lim_{x\to\infty} f'(x) = 0$ and $f(0) = 0$.

This assumption states that the benefit is always nonnegative and also monotonically increases with the consumption $r^{(t)}$, whereas the marginal benefit monotonically decreases against $r^{(t)}$. Such assumption widely adopted in previous works, e.g. [20].

The consumer's cost in each time slot includes two parts. The first part is the total expense on electricity purchase, which is $q^{(t)} a^{(t)}$. The second part is the cost for electricity storage, which equals to $cb^{(t+1)}$, where c is a constant representing the unit storage cost.[6]

Combining the benefit and costs, the consumer's one-slot utility u is:

$$u(s^{(t)}, a^{(t)}) = f(r^{(t)}) - q^{(t)} a^{(t)} - cb^{(t+1)}. \tag{3}$$

The consumer's storage management policy in the MDP is a mapping from the state to its action: $\pi : \mathcal{Q} \times \mathcal{B} \times \mathcal{E} \to \mathcal{A}$. That is, given a state s, this policy instructs the consumer to take the action $a = \pi(s)$. In the rest of this paper, we focus on

[5]It should be noted that the Markovian price model assumed here is only for analytical tractability. We show in Section V that our proposed storage management algorithm also performs well when the price variation is not Markovian.

[6]The linear cost model for electricity storage has been adopted in many existing works, e.g. [20]. Also, we would like to note that our proposed learning algorithm can be easily extended to apply to other cost models for energy storage. For example, the electricity storage cost could also be formulated as a convex function [31] or a piece-wise linear function. Since the MDP formulation and all our current analysis still apply with such non-linear storage costs, the proposed online learning algorithm is also capable of learning the optimal storage management policy.

optimizing the policy to maximize the consumer's expected long-term utility, which is referred to as the *value function* and defined as the expectation of the discounted sum of the consumer's one-slot utility:

$$U^{(\pi)}(s^{(0)}) = \mathbb{E}\left(\sum_{t=0}^{\infty} \delta^t u(s^{(t)}, a^{(t)}) \mid s^{(0)}\right). \tag{4}$$

Here $\delta < 1$ is a constant discount factor, which represents the fact that the consumer puts a higher weight on its current utility than its utilities in the future.[7]

Given 4, the optimization problem is formalized as follows:

$$\max_{\pi} U^{\pi}(s^{(0)}). \tag{5}$$

From [25], it is known that in an MDP, the problem 5 is equivalent to the following optimization:

$$\max_{\pi} U^{\pi}(s), \ \forall s \in \mathcal{S}. \tag{6}$$

Let π^* and $\{U^*(s)\}$ denote the solution of 6 and the corresponding optimal long-term utility respectively, it is well-known that π^* and $U^*(s)$ can be obtained by recursively solving the following Bellman equation set [25]:

$$U^*(s) = \max_{a \in \mathcal{A}} \left(u(s,a) + \delta \sum_{s' \in \mathcal{S}} p(s'|s,a) U^*(s')\right). \tag{7}$$

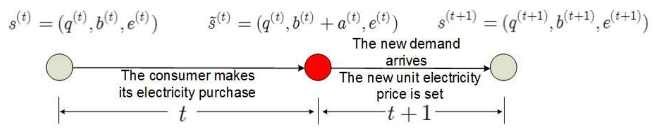
In the next section, we solve this Bellman equation set using the idea of dynamic programming and online learning.

III. Post-Decision State Based Dynamic Programming

In this section, we analyze and solve the Bellman equation 7. The traditional algorithms for solving the Bellman equation, e.g. the value iteration and the policy iteration [25], need the state transition probability and the state space to be known a priori, and thus are not feasible solutions for our problem since these values are not known (or only partially known) a priori. To this end, we propose an online reinforcement learning algorithm to dynamically learn the state transition probability and the state space without any prior knowledge in order to solve π^* and U^* on-the-fly.

The rest of this section is organized as follows. We first introduce the concept of the post-decision state (PDS) in Section III-A. Section III-B then develops a general PDS based online learning algorithm that allows the consumer to integrate known information about the system dynamics into its learning process. By exploiting the partially known information about the system dynamics, the PDS based learning algorithm significantly improves its convergence speed and run-time performance compared to the conventional online learning algorithms, e.g. Q-learning [25]. Finally in Section III-C, we prove the convergence of the proposed algorithm.

[7]It should be noted that we only consider stationary policy in this paper, which is common for most MDP problems[25]. Therefore, the mapping π does not change over time. But the action $a^{(t)}$ adopted by the consumer changes over time depending on the state $s^{(t)}$.


$s^{(t)} = (q^{(t)}, b^{(t)}, e^{(t)})$
$\tilde{s}^{(t)} = (q^{(t)}, b^{(t)} + a^{(t)}, e^{(t)})$
$s^{(t+1)} = (q^{(t+1)}, b^{(t+1)}, e^{(t+1)})$
The consumer makes its electricity purchase
The new demand arrives
The new unit electricity price is set
t
$t+1$
Fig. 3. Illustration of the post-decision state

A. Post-Decision State

The most critical idea in our proposed learning algorithm is to introduce an intermediate state in order to capture the known part of the system dynamics in each time slot and speed-up the learning process. We call this intermediate state the post-decision state (PDS) $\tilde{s} \triangleq (\tilde{q}, \tilde{b}, \tilde{e})$. In a brief explanation, the PDS represents the state of the system in each time slot after the consumer performs its action $a^{(t)}$ but before the demand $d^{(t)}$ is realized. The relationship between a state $s^{(t)}$ and its PDS $\tilde{s}^{(t)}$ in time slot t is illustrated in Figure 3. From this, the corresponding PDS in the time slot t is computed as follows:

$$\tilde{q}^{(t)} = q^{(t)}, \tilde{b}^{(t)} = b^{(t)} + a^{(t)}, \tilde{e}^{(t)} = e^{(t)} \tag{8}$$

Accordingly, we define the post-decision value function $V^*(\tilde{s})$ for a PDS $\tilde{s}$ as follows: [8]

$$V^*(\tilde{s}) = \sum_{s' \in \mathcal{S}} p(s'|s,a) U^*(s'). \tag{9}$$

For the better illustration, we refer to s as the "normal" state and $U^*(s)$ as the "normal" value function, in order to differentiate with their post-decision counter parts.

Compare 7 and 9, it can be noticed that the post-decision value function represents the expectation of the consumer's future utilities over the unknown system dynamics. Hence, there is a deterministic mapping from the normal value function to the post-state value function. By substituting 9 into 7, the relationship between the normal value function and the post-state value function can be expressed as follows:

$$U^*(s) = \max_{a \in \mathcal{A}} \left(u(s,a) + \delta V^*(\tilde{s})\right). \tag{10}$$

The above equation shows that the normal value function $U^*(s)$ at each time slot is obtained from the corresponding post-decision value function $V^*(\tilde{s})$ at the same time slot, where $\tilde{s} = (q, b+a, e)$, by performing the maximization over the action a.

The advantages of introducing the post-decision state and corresponding value functions are summarized next.

- In the normal state based Bellman's equation set 7, the expectation over the possible environment dynamics e, the possible demand arrival d and the possible energy prices q has to be performed before the maximization over the possible energy purchasing actions a. Hence, performing the maximization requires the knowledge of these dynamics. In contrast in the post-decision state-based Bellman's equations 9 and 10, computing this expectation is separated from the maximization, which can be parallel. Therefore if we can directly learn and

[8]In our subsequent analysis, we drop the time index (t) from the expressions, e.g. in 9, when the properties being discussed apply to all states and actions. The time index will be kept only when necessary.

- approximate the post-decision value function $V^*(\tilde{(s)})$ online, the maximization (a.k.a. the optimization of the storage management policy) can be solved without any prior knowledge of the system dynamics.
- Given the action a, the post-decision state factors the (Markovian) system dynamics into an a priori unknown component, i.e. e, d and q whose evolution is independent of a, and an a priori known component, i.e. the storage b whose evolution is determined by a. It is important to note that the evolution of the a priori unknown component is independent to b as well. This fact enables us to develop a batch update scheme on the post-decision value functions, which is discussed in the next section. Such batch update scheme can significantly improve the convergence speed of the proposed PDS based learning algorithm, compared to conventional online learning algorithms.

B. Post-Decision State Based Online Learning

In this section, we propose the PDS based online learning algorithm that utilizes adaptive approximation to effectively learn the post-state value functions $\{V^*\tilde{s}\}$. It is known from 10 that we can also obtain $\{U^*(s)\}$ once $\{V^*\tilde{s}\}$ are computed. In each time slot t, the post-decision value function is updated in the following manner:

$$V^{(t)}(\tilde{s}^{(t)}) = (1-\alpha^{(t)})V^{(t-1)}(\tilde{s}^{(t)}) + \alpha^{(t)}U^{(t)}(s^{(t)}). \quad (11)$$

Here $\alpha^{(t)}$ is the learning rate factor that satisfies $\sum_{t=0}^{\infty}\alpha^{(t)} = \infty$ and $\sum_{t=0}^{\infty}(\alpha^{(t)})^2 < \infty$, e.g. $\alpha^{(t)} = \frac{1}{t}$. $U^{(t)}(s^{(t)})$ is the normal value function updated in time slot t, which is computed as follows:

$$U^{(t)}(s^{(t)}) = \max_{a\in\mathcal{A}}\left(u(s^{(t)},a) + \delta V^{(t-1)}(\tilde{s}^{(t)})\right). \quad (12)$$

Remark: With 11 and 12, the normal value function and the post-decision value function are updated iteratively in each time slot. In the first step, the normal value function of the current state $s^{(t)} = (q^{(t)}, b^{(t)}, e^{(t)})$ is updated to $U^{(t)}(s^{(t)})$ using the (un-updated) post-decision value function $V^{(t-1)}(\tilde{s}^{(t)})$ where $\tilde{s}^{(t)} = (q^{(t)}, b^{(t)} + a^{(t)}, e^{(t)})$. In the second step, the post-decision value function of $\tilde{s}$ is updated to $V^{(t)}(\tilde{s}^{(t)})$ using the updated normal value function $U^{(t)}(s^{(t)})$. In the next section, we prove that such iterative update process introduced by 11 and 12 ensures both the normal value function and the post-decision value function converge to their optimal values, i.e. $\{U^*(s)\}$ and $\{V^*(\tilde{s})\}$.

The above iterative update process 11 and 12, though ensures the convergence to the optimal value, only updates in each time slot t the currently visited PDS $\tilde{s}^{(t)}$. Noting that the temporal transition of q, e and d are all independent to the electricity storage b, or in other words, the values of $e^{(t)}$, $q^{(t)}$ and $d^{(t)}$ can be realized with any possible value of b. Therefore instead of solely updating one PDS $\tilde{s}^{(t)}$ in a time slot t, we can perform a batch update over any PDS $\tilde{s} = (\tilde{q}, \tilde{b}, \tilde{e})$, which satisfies that $\tilde{q} = \tilde{q}^{(t)}$ and $\tilde{e} = \tilde{e}^{(t)}$, as shown below:

$$\begin{aligned} V^{(t)}(\tilde{q}^{(t)}, \tilde{b}, \tilde{e}^{(t)}) &= (1-\alpha^{(t)})V^{(t-1)}(\tilde{q}^{(t)}, \tilde{b}, \tilde{e}^{(t)}) \\ &+\alpha^{(t)}U^{(t)}(\tilde{q}^{(t)}, \tilde{b} - a^{(t)}, \tilde{e}^{(t)}), \forall \tilde{b} \in \mathcal{B}. \end{aligned} \quad (13)$$

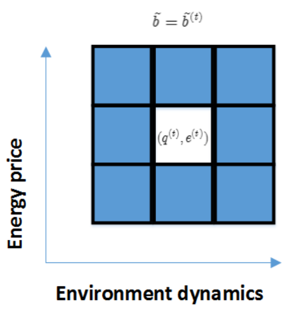


Fig. 4. The value function update in conventional reinforcement learning.

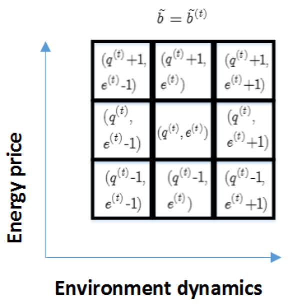


Fig. 5. The batch value function update in PDS based learning.

Algorithm 1 PDS-based online learning

```
Initialize: $V^{(0)}(\tilde{s}) = 0$ for all $\tilde{s} \in \mathcal{S}$; $t = 1$.
Repeat
(1) Update the normal state $s^{(t)} = (q^{(t)}, b^{(t)}, e^{(t)})$ with $b^{(t)} = \max\{\tilde{b}^{(t-1)} - d^{(t-1)}, 0\}$;
(2) Compute the optimal action $a^{(t)}$ for the current normal state $s^{(t)}$ and update $U^{(t)}(s^{(t)})$ as in Eq.12;
(3) Batch update the post-decision value functions $V^{(t)}(\tilde{q}^{(t)}, \tilde{b}, \tilde{e}^{(t)}), \forall \tilde{b} \in \mathcal{B}$ as in Eq. 13;
(4) Update the PDS $\tilde{s}^{(t)} = (q^{(t)}, \tilde{b}^{(t)}, e^{(t)})$ with $\tilde{b}^{(t)} = b^{(t)} + a^{(t)}$;

$t := t + 1$
End
```

With the batch update 13, we are able to update all the PDSes $\{(q^{(t)}, \tilde{b}, e^{(t)}), \forall \tilde{b} \in \mathcal{B}\}$ all at once in time slot t, as shown in Figure 5, instead of updating only one PDS $\tilde{s}^{(t)}$ as shown in Figure 4. Here the white cells represent the PDSes whose value functions are updated in the current time slot and the blue cells represent the PDSes whose value functions are not updated in the current time slot. As a result, the convergence speed of our proposed learning algorithm is significantly improved, which will be illustrated in Section V.

In summary, our proposed PDS-based load scheduling algorithm is illustrated in Algorithm 1.

C. Convergence of the PDS Learning Algorithm

In this section, we analyze the convergence property of the PDS learning algorithm, which is proven in the following theorem.

Theorem 1. The PDS based online learning algorithm converges to the optimal post-decision value function $\{V^*(\tilde{s})\}$ when the sequence of learning rates $\alpha^{(t)}$ satisfies $\sum_{t=0}^{\infty}\alpha^{(t)} = \infty$ and $\sum_{t=0}^{\infty}(\alpha^{(t)})^2 < \infty$.

Proof: For each PDS $\tilde{s}$, we define

$$F_{\tilde{s}}(V) \triangleq \max_{a\in\mathcal{A}}(u(s,a)+\delta V(\tilde{s})), \tag{14}$$

where s and $\tilde{s}$ satisfy the relationship that $q=\tilde{q}$, $e=\tilde{e}$, and $b=\tilde{b}-a$.

We also define $F:\mathbb{R}^{|\mathcal{S}|}\to\mathbb{R}^{|\mathcal{S}|}$ be a mapping such that $F(V)=[F_{\tilde{s}}(V)]_{\tilde{s}}$. [28] has proven that the convergence of our proposed algorithm is equivalent to the convergence of the associated O.D.E.:

$$\dot{V}=F(V)-V. \tag{15}$$

Since the map $F:\mathbb{R}^{|\mathcal{S}|}\to\mathbb{R}^{|\mathcal{S}|}$ is a maximum norm δ-contraction [30], the asymptotic stability of the unique equilibrium point of the above O.D.E. is guaranteed in [30]. This unique equilibrium point corresponds to the optimal post-decision state value function $\{V^*(\tilde{s})\}$. ■

Theorem 1 shows that Algorithm 1 converges to the optimal post-decision value function $\{V^*(\tilde{s})\}$. Since the optimal normal value function $\{U^*(s)\}$ is a deterministic function of $\{V^*(\tilde{s})\}$, it is thus concluded that Algorithm 1 also converges to $\{U^*(s)\}$ and the optimal policy π^*. Therefore, we prove that the consumer is able to learn the optimal storage management policy through Algorithm 1.

IV. Optimal Scheduling with Deferrable Loads

In the analysis so far, we assume that the load demands of each consumer are non-deferrable. That is in a time slot t, if the available amount of energy, i.e. $b^{(t)}+a^{(t)}$, is lower than the current load demand $d^{(t)}$, the consumer cannot benefit by deferring the deficit $d^{(t)}-b^{(t)}-a^{(t)}$ to be fulfilled in future time slots. However, in many smart grid applications, the load demands from various appliances are deferrable, e.g. electrical vehicles, dish washers, washers/dryers [1]. A common feature of these appliances is that their loading cycles are long while their starting time can be easily shifted. In this section, we particularly analyze how to perform the dynamic load scheduling on such deferrable load demands using the idea of PDS-based learning.

A. MDP Formulation for the Scheduling with Deferrable Loads

In order to differentiate with the scenario of non-deferrable loads, we introduce an additional variable $y^{(t)}$ representing the amount of demand load from the previous time slot that is left unfulfilled. The total load of the consumer in time slot t is denoted by $h^{(t)}=d^{(t)}+y^{(t)}$. We also have

$$y^{(t)}\triangleq\max\{0,h^{(t-1)}-b^{(t-1)}-a^{(t-1)}\}. \tag{16}$$

To solve the optimal storage management policy with deferrable loads, we also propose an MDP to formulate the consumer's decision problem. The MDP proposed in this section is referred to as the *MDP with deferrable loads*, with all the associated variables subscripted by dl, while we refer to the MDP proposed in Section II as the *MDP with non-deferrable loads*.

Given 16, the MDP with deferrable loads can be easily extended based on the MDP with non-deferrable loads, by incorporating the deficit into the state definition, which is specified as follows:

- The *state* is defined as a tuple $s_{dl}^{(t)}\triangleq(q^{(t)},b^{(t)},e^{(t)},y^{(t)})$. Therefore, the consumer's storage managements decision in each time slot is also influenced by the deficit $y^{(t)}$ from the previous period. It should be noted that since both $d^{(t)}$ and $b^{(t)}$ all take values from finite sets, $y^{(t)}$ also takes values from a finite set, which is denoted by $\mathcal{Y}$.
- The action is still defined as the consumer's electricity purchase in each period, i.e. $a_{dl}^{(t)}\triangleq a^{(t)}$.
- Since $y^{(t)}$ is a deterministic function of $y^{(t-1)}$, $b^{(t)}$, $a^{(t)}$ and $d^{(t)}$, as shown in 16, the state transition probability is

$$\begin{aligned}&p(s_{dl}^{(t+1)}|s_{dl}^{(t)},a_{dl}^{(t)})\\&=p_q(q^{(t+1)}|q^{(t)},e^{(t)})p_e(e^{(t+1)}|e^{(t)})p_d(d^{(t)}|e^{(t)})\\&I\left(b^{(t+1)}=\min\{\max\{b^{(t)}+a^{(t)}-h^{(t)},0\},B\}\right)\\&I\left(y^{(t+1)}=\max\{0,h^{(t)}-b^{(t)}-a^{(t)}\}\right).\end{aligned} \tag{17}$$

It is important to note that we do not impose the expiration deadline for the deferrable loads, as assumed in many existing works such as [18]. As a result, the demanded loads deferred from different time slots provide equal unit benefit to the consumer when being fulfilled. However, we will show in Section VI-B that the proposed MDP formulation in this section can be easily extended to the scenario where the load demands have (heterogeneous) expiration deadlines, by incorporating the deadlines into the definition of the state $s_{dl}^{(t)}$.

Define $r_{dl}^{(t)}\triangleq\min\{b^{(t)}+a^{(t)},h^{(t)}\}$, the one-slot utility u_{dl} can be expressed as follows:

$$u_{dl}(s_{dl}^{(t)},a_{dl}^{(t)})=f(r_{dl}^{(t)})-q^{(t)}a^{(t)}-cb^{(t+1)}. \tag{18}$$

Finally, the consumer's expected long-term utility of the consumer can be expressed as

$$U_{dl}^{(\pi)}(s_{dl}^{(0)})=\mathbb{E}\left(\sum_{t=0}^{\infty}\delta^t u_{dl}(s_{dl}^{(t)},a_{dl}^{(t)})\mid s_{dl}^{(0)}\right). \tag{19}$$

It is important to note that since the discount factor $\delta<1$, the consumer's benefit received from one unit load monotonically decreases against time. Therefore, deferring the demand load decreases the consumer's long-term utility 19. This assumption reflects the negative effect of load deferment, which is also referred as the "inconvenience cost" caused by the delay introduced in the energy usage [1].

B. PDS-based Learning Algorithm for the MDP with Deferrable Loads

In this section, we propose the PDS-based learning algorithm to solve the MDP with deferrable loads. Similar to 8, the PDS $\tilde{s}_{dl}^{(t)}\triangleq\{\tilde{q}_{dl}^{(t)},\tilde{b}_{dl}^{(t)},\tilde{e}_{dl}^{(t)},\tilde{y}_{dl}^{(t)}\}$ represents the state of the system in each time slot after the consumer performs its action $a_{dl}^{(t)}$ but before the demand $d^{(t)}$ arrives. Thus we have

$$\tilde{q}_{dl}^{(t)}=q^{(t)},\tilde{b}_{dl}^{(t)}=b^{(t)}+a_{dl}^{(t)},\tilde{e}_{dl}^{(t)}=e^{(t)},\tilde{y}_{dl}^{(t)}=y^{(t)}. \tag{20}$$

Algorithm 2 PDS-based online learning with deferrable loads

```
Initialize: $V^{(0)}(\tilde{s}) = 0$ for all $\tilde{s_{dl}}$; $t = 1$.
Repeat
(1) Update the normal state $s_{dl}^{(t)} = (q^{(t)}, b^{(t)}, e^{(t)}, y^{(t)})$ with $b^{(t)} = \tilde{b}_{dl}^{(t-1)} - d^{(t-1)} - y^{(t-1)}$ and $y^{(t)} = \max\{0, y^{(t-1)} + d^{(t-1)} - b^{(t-1)} - a_{dl}^{(t-1)}\}$;
(2) Compute the optimal action $a_{dl}^{(t)}$ for the current normal state $s_{dl}^{(t)}$ and update $U_{dl}^{(t)}(s_{dl}^{(t)})$ as in Eq.21;
(3) Batch update the post-decision value functions $V_{dl}^{(t)}(\tilde{q}^{(t)}, \tilde{b}, \tilde{e}^{(t)}, \tilde{y}^{(t)}), \forall \tilde{b} \in \mathcal{B}$;
(4) Update the PDS $\tilde{s}_{dl}^{(t)} = (q^{(t)}, \tilde{b}^{(t)}, e^{(t)})$ with $\tilde{b}^{(t)} = b^{(t)} + a_{dl}^{(t)}$;

$t := t + 1$
End
```

Correspondingly, the post-decision value function is updated as

$$V_{dl}^{(t)}(\tilde{s}_{dl}^{(t)}) = (1 - \alpha^{(t)})V_{dl}^{(t-1)}(\tilde{s}_{dl}^{(t)}) + \alpha^{(t)}U_{dl}^{(t)}(s_{dl}^{(t)}). \quad (21)$$

and the normal value function is updated as

$$U^{(t)}(s_{dl}^{(t)}) = \max_{a_{dl} \in \mathcal{A}} \left(u_{dl}(s_{dl}^{(t)}, a_{dl}) + \delta V_{dl}^{(t-1)}(\tilde{s}_{dl}^{(t)}) \right). \quad (22)$$

Given the PDS, the online learning algorithm for the MDP with deferrable loads is illustrated in Algorithm 2.

Similar to Theorem 1, we have that Algorithm 2 also converges, as shown in the following theorem.

Theorem 2. Algorithm 2 converges to the optimal post-decision value function $\{V_{dl}^*(\tilde{s}_{dl})\}$ when the sequence of learning rates $\alpha^{(t)}$ satisfies $\sum_{t=0}^{\infty} \alpha^{(t)} = \infty$ and $\sum_{t=0}^{\infty} (\alpha^{(t)})^2 < \infty$. ■

In the next theorem, we prove that Algorithm 2 always outperforms Algorithm 1.

Theorem 3. Given two initial states $s^{(0)} = (q^{(0)}, b^{(0)}, e^{(0)})$ and $s_{dl}^{(0)} = (q^{(0)}, b^{(0)}, e^{(0)}, 0)$, the inequality $U_{dl}^*(s_{dl}^{(0)}) \geq U^*(s^{(0)})$ always holds. ■

Theorem 3 proves that the consumer always obtains a higher long-term utility when the loads are deferrable. Therefore, the optimal long-term utility achieved in the MDP with deferrable loads is always no less than the optimal long-term utility achieved in the MDP with non-deferrable loads.

V. Illustrative Examples

In this section, we provide numerical results to illustrate the performance of our proposed online learning algorithm. We consider a power grid with 100 consumers, where the length of each time slot is 1 hour. The energy storage capacity $B = 10kWh$. The environment dynamics e represents the hour that each time slot is located in a day and hence, we have $\mathcal{E} = \{0, 1, \ldots, 23\}$ and $e^{(t)} = \mod(t, 24)$.

In one day, we divide the time slots into peak and non-peak hours. Specifically, the peak hours are 6pm to 12am and the remaining hours are non-peak hours. The demand of each consumer in each time slot follows a truncated Gaussian distribution in the region $[0, 2.5kWh]$ as follows

$$p_d(d^{(t)} \mid e^{(t)}) = \begin{cases} \mathcal{N}(0.5, 0.2^2), \text{if } e^{(t)} \in [0, 17] \\ \mathcal{N}(1, 0.1^2), \text{if } e^{(t)} \in [18, 23] \end{cases} \quad (23)$$

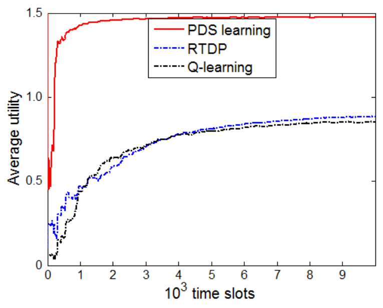


Fig. 6. Run-time performances of online learning algorithms

The unit electricity price is taken from a finite set $\mathcal{Q} = \{0.1, 0.2, \ldots, 0.5\}$. We also set $c = 0.1$ and the benefit function to be a logarithmic function $f(x) = log(1 + x)$ as in [17].

A. Experiments with Non-deferrable Loads

We first conduct experiments in power grid systems where the demand loads are non-deferrable. In the first experiment, we compare our algorithm (i.e. Algorithm 1) with three state-of-the-art online learning algorithms to illustrate the advantage of introducing the PDS and batch update:

(1) Value iteration [25] is an off-line algorithm, which we are solely using here as the optimal benchmark. This algorithm needs the complete knowledge of the underlying state transition probabilities and utility functions. The computation complexity of value iteration is also significantly higher than online reinforcement learning methods.

(2) Q-learning [25] is a model-free reinforcement learning algorithm. It does not require a priori knowledge of the state transition probabilities and utility functions, but suffers from slow convergence speed.

(3) Real-time dynamic programming (RTDP) [29] is also a model-based online learning algorithm. When implementing RTDP, the learning agent first constructs a statistic model of the underlying MDP and then updates the state transition probabilities in this statistic model using its past experiences. Therefore, the state space of the MDP needs to be known a priori.

Table II shows the average performance received by the four learning algorithms in the considered power grid.[9] It can be observed that PDS learning significantly outperforms the other two online learning algorithms (RTDP and Q-learning) on all metrics (higher energy consumption per consumer and lower cost for unit energy per consumer). As a result, the average one-slot utility achieved by PDS learning is close to the optimal value achieved by the off-line value iteration algorithm.

Figure 6 plots the run-time performances of the online learning algorithms across 10000 time slots. Note that PDS learning converges after 1122 time slots (with the run-time average utility achieving 90% of the highest value), while

[9]For the online algorithms (PDS learning, Q-learning, RTDP), we run each of them for 10000 time slots. For the off-line value iteration, we run the algorithm until it converges (because there is no intermediate output for the value iteration before its convergence).

TABLE II
COMPARISON WITH OTHER LEARNING ALGORITHMS

Per time slot	*Value iteration*	*RTDP*	*Q-learning*	*PDS learning*
Average utility	1.4320	0.9047	0.9987	1.3394
Average consumed energy	0.7746 kWh	0.4272 kWh	0.4537 kWh	0.7847 kWh
Average purchased price	\$0.2221/kWh	\$0.2930/kWh	\$0.2928/kWh	\$0.2914/kWh

TABLE III
COMPARISON WITH OTHER STORAGE MANAGEMENT ALGORITHMS

Per time slot	*Price prediction*	*Load prediction*	*Price & load prediction*	*PDS learning*
Average utility	0.4432	0.3654	0.4565	1.3394
Average consumed energy	0.3909 kWh	0.3998 kWh	0.3098 kWh	0.7847 kWh
Average purchased price	\$0.5349/kWh	\$0.6729/kWh	\$0.7095/kWh	\$0.2914/kWh

RTDP converging after 2983 time slots and Q-learning converging after 3132 time slots. Also, the average one-slot utilities achieved by RTDP and Q-learning upon convergence are significantly worse than that achieved by PDS, which indicates that both RTDP and Q-learning are not able to learn the optimal storage management policy.

In the experiments so far, we do not assume any prior knowledge is known by the consumer about the system dynamics, before it learns the optimal policy. That is, the consumer has no knowledge about the transition probabilities $p_q(q' \mid q, e)$, $p_e(e' \mid e)$, and $p_d(d \mid e)$, as well as the state space $\mathcal{Q}$, $\mathcal{D}$ and $\mathcal{B}$, at the beginning of the experiment, and has to learn these values on-the-fly. Therefore, the convergence speed shown in Figure 6 represents the performance of our PDS learning algorithm in the worse-case scenario, and thus can be viewed as the upper-bound of the achivable convergence speed of PDS learning.

Nevertheless, such "zero-prior" assumption usually does not hold in practice, as the consumer always has some prior knowledge about the smart grid system and the underlying MDP, which can help the consumer refine its initial estimation of the value function and the storage management policy. In the next experiment, we examine how fast PDS learning converges with slightly more knowledge on the system.

We conduct two experiments on this. In the first case, we assume that the consumer knows the structure of the spaces $\mathcal{Q}$, $\mathcal{D}$ and $\mathcal{B}$, as well as $p_e(e' \mid e)$ (i.e. the consumer knows the hour that the current time slot resides in). The learning performance under this setting is shown in Figure 7. PDS learning algorithm still converges significantly faster than other reinforcement learning algorithms in this case. Importantly, PDS learning converges after only 50 time slots, which is significantly faster than the speed it achieves when no prior about the MDP is known and utilized.

In the second case, we go one step further to see whether the convergence speed can be further improved if the consumer knows the dynamics of the electricity price as well, i.e. $\{p_q(q^{(t+1)}) \mid q^{(t)}, e^{(t)}\}$. This is a reasonable assumption since today's ISOs can perform a good price forecast, by utilizing

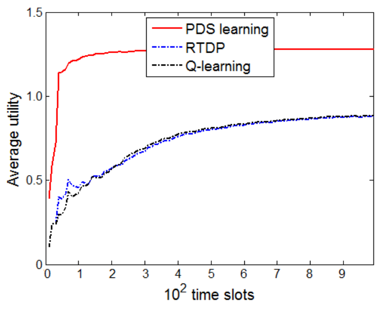


Fig. 7. Run-time performances of online learning algorithms with prior knowledge

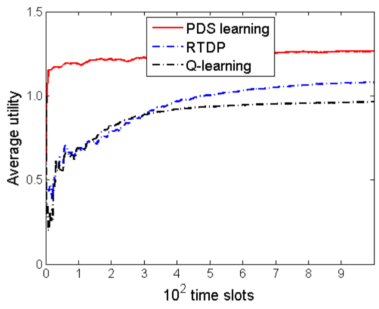


Fig. 8. Run-time performances of online learning algorithms with prior knowledge on electricity price

the large amount of historical data. From Figure 8, we can see the convergence speed of PDS learning is further improved. It converges after 12 time slots (by achieving an average utility which is 95% of the optimal value).

Next, we compare Algorithm 1 against the existing price-aware energy storage management algorithms based on load and price forecasts, which are discussed in [12]-[14]. The algorithms in these three works function under the same principle: each consumer tries to optimize its energy purchases in each hour in order to maximize its next-hour profit from energy consumption, based on its prediction over the next-hour load demand and energy price. Meanwhile, all these algorithms assume that each consumer presumes a model on

TABLE IV
ENERGY PRICE FOR CPP CUSTOMERS

Summer off-peak	*Summer critical peak*	*Winter off-peak*	*Winter critical peak*
12.3c/kWh	77.1c/kWh	57.5c/kWh	49.8c/kWh

TABLE V
PERFORMANCE OF STORAGE MANAGEMENT ALGORITHMS WITH THE REAL ENERGY PRICE DATA

Per time slot	*Price prediction*	*Load prediction*	*Price & load prediction*	*PDS learning*
Average utility	0.5549	0.7773	0.4565	1.2135
Average consumed energy	0.4430 kWh	0.6821 kWh	0.4352 kWh	1.2343 kWh
Average purchased price	\$0.4242/kWh	\$0.3232/kWh	\$0.4201/kWh	\$0.3309/kWh

TABLE VI
PERFORMANCE ACHIEVED BY DIFFERENT ALGORITHMS WITH DEFERRABLE LOADS

Per time slot	*Value iteration*	*RTDP*	*Q-learning*	*PDS learning*
Average utility	2.5074	2.2109	1.7772	1.4444
Average consumed energy	2.1897 kWh	0.9723 kWh	0.9025 kWh	1.8838 kWh
Average purchased price	\$0.1825/kWh	\$0.2514/kWh	\$0.2502/kWh	\$0.2013/kWh

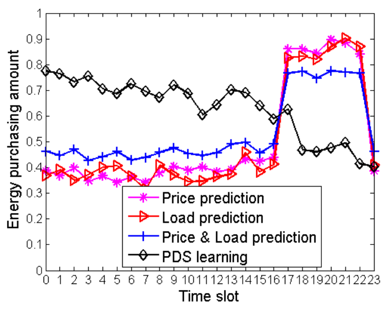


Fig. 9. The consumer's per-slot electricity purchase

the variation of the load demand and price and makes its prediction based on this presumption. With such a principle, we develop three new algorithms for our storage management problem.[10]

(1) **Price Prediction**. This algorithm is similar to the algorithm in [12], which predicts the variation of the next-hour energy price while neglecting the variation of the load demand. Specifically, it assumes that the next-hour load demand is the same as the demand that the consumer observes today.

(2) **Load Prediction**. This algorithm follows the idea of the algorithm in [14], which predicts the variation of the next-hour load demand and assumes that the next-hour energy price is the same as the price of today.

(3) **Price & Load Joint Prediction**. This algorithm follows the idea of the algorithm in [13] and predicts the variations of both the next-hour load demand and energy price using a known statistical model.

We then plot in Figure 9 the consumer's per-slot electricity purchase under different algorithms. Here we assume that when the consumer knows the statistical distribution of the next-hour energy price as well as the next-hour demand load, i.e. $\{p_e(e^{(t+1)} \mid e^{(t)})\}$, $\{p_q(q^{(t+1)} \mid q^{(t)}, e^{(t)})\}$, and $p_d(d^{(t)} \mid e^{(t)})$, when making its prediction over these dynamics. It can be observed that the three benchmark algorithms delivers low per-slot purchase during off-peak hours and high purchase during peak hours, which is consistent with the fluctuation of the consumer's load demand. This is because these three algorithms only focus on maximizing the consumer's utility in the next hour while neglecting the impact of its current decision over the further future. As a result, these three algorithms can only optimize one hour ahead, under which the consumer starts to increase its purchase at 17:00pm (1 hour before the peak hours) and reduce its purchase at 23:00pm (1 hour before the off-peak hours). These algorithms thus do not fully exploit the information embedded in the price and load fluctuation to effectively purchase and store energy during the off-peak hours when the energy price is low. In contrast, Algorithm 1 instructs the consumer to purchase and store sufficient amounts of energy during off-peak hours (when the energy price is low) for the consumption during peak hours (when the energy price is high). Therefore, the average purchased prices of the unit amount of energy are significantly higher under the benchmark algorithms than that under Algorithm 1, which results in a significantly lower average utility, as shown in III.

We also run a separate experiment on the real energy price data set to test the efficacy of our algorithm. The pricing data we used is from the Critical Peak Pricing (CPP) plan in the PowerCentsDC program [22]. In the CPP plan, the

[10]It should be noted here that in [12]-[14], the decision variable to be optimized is not the purchased amount of energy in each time slot. Hence, the three algorithms proposed here, i.e. price prediction, load prediction, price & load prediction, are not exactly the same as those in [12]-[14], but share the same design principle with them (i.e. optimizing one-hour-ahead utility based on pre-established statistical models).

energy price is higher in the critical peak hours than that in the normal (off-peak) hours. The critical peak hours occur between 2pm to 6pm in the summer and between 6am to 8am and 6pm and 8pm during the winter months. The energy price is shown in Table IV. Using this pricing plan, the results of the learning algorithms are shown in Table V, from which we can observe that the PDS based learning algorithm still significantly outperforms the others.

B. Experiments with Deferrable Loads

This section evaluates the performance of PDS learning in power grid systems where the demand loads are deferrable. Table VI shows the average performance received by an individual consumer when running Algorithm 2 and the three benchmark learning algorithms discussed in Section V-A. Compared to the results in Table II, it should be noted that the performances of all algorithms (measured by the average utility) significantly increase when the demand are deferrable, as proven in Theorem 3. Meanwhile, the average electricity consumed in each time slot also increases, since the consumer's unfulfilled demand will be deferred into subsequent time slots. It is also interesting to note that the averaged purchased price for unit energy does not significantly change in this scenario. Therefore, the increase of the consumer's utility mainly comes from the increased (and more effective) energy consumption.

VI. Extensions

In this section, we discuss various extensions of our proposed framework and the PDS online learning algorithms. For each extension, the discussion here is limited to preliminary modelling and numeric experiments. The detailed analysis is relegated as important future works.

A. Dynamics of Renewable Generation

In the current smart grid systems, the uncertainty also lies in the renewable generation, as their capacity (e.g. wind and solar) is highly dynamic and non-dispatchable. In this section, we discuss how our framework can be extended to incorporate renewable generation dynamics.

Let $G^{(t)}$ represent the amount of electricity generated in time slot t, i.e. the total amount of electricity available to consume in this time slot. When a consumer makes a electricity purchase $a^{(t)}$, we assume that the actual amount he gets to consume is determined by a function $\phi(G^{(t)}, a^{(t)}) \leq a^{(t)}$. The function phi monotonically increases with $G^{(t)}$ and $a^{(t)}$. That is, a consumer can get sufficient electricity to consume only if the generation $G^{(t)}$ is sufficiently large. When $G^{(t)}$ is small, i.e. there is certain energy shortage, a consumer will get less electricity than it requests. To quantify the stochastic behavior of the renewable generation, we also assume that the evolution of $G^{(t)}$ follows a conditional probability distribution $\{p_G(G^{(t)} \mid e^{(t)})\}$.

Given the Markovian evolution of $e^{(t)}$, the stochastic control problem can still be casted as a Markov Decision Process defined as follows:

- The state is defined as $s^{(t)} \triangleq (q^{(t)}, b^{(t)}, e^{(t)}, G^{(t)})$.
- The state transition probability incorporates the dynamics of $G^{(t)}$, which is expressed as follows:

$$\begin{aligned} &p(s^{(t+1)}|s^{(t)}, a^{(t)}) = p_e(e^{(t+1)}|e^{(t)}) p_d(d^{(t)}|e^{(t)}) \\ &p_q(q^{(t+1)}|q^{(t)}, e^{(t)}) p_G(G^{(t)} \mid e^{(t)}) \\ &I\left(b^{(t+1)} = \min\{\max\{b^{(t)} + a^{(t)} - d^{(t)}, 0\}, B\}\right), \end{aligned} \tag{24}$$

- The consumer's one-slot utility is

$$u(s^{(t)}, a^{(t)}) = f(r^{(t)}) - q^{(t)} a^{(t)} - c b^{(t+1)}, \tag{25}$$

where $r^{(t)}$ is redefined as $\min\{b^{(t)} + \phi(G^{(t)}, a^{(t)}), d^{(t)}\}$.

Next we conduct an experiment to illustrate how PDS learning performs under the dynamics of renewable generation. Specifically, we assume that $G^{(t)}$ follows a truncated Gaussian distribution in the region $[100kWh, 500kWh]$:

$$p_G(G^{(t)} \mid e^{(t)}) = \begin{cases} \mathcal{N}(300, 0.25^2), \text{if } e^{(t)} \in [0, 17] \\ \mathcal{N}(400, 0.15^2), \text{if } e^{(t)} \in [18, 23] \end{cases} \tag{26}$$

The function phi is defined as follows:

$$\phi(G^{(t)}, a^{(t)}) = \begin{cases} a, \text{if } G \geq 400kWh \\ \dfrac{G}{400} a, \text{if } G < 400kWh \end{cases} \tag{27}$$

Table VII shows the performance of various algorithms. Compared to Table II and III, we can observe that the average utility achieved by all algorithms significantly decrease because of the dynamic generation and sporadic energy shortage. However, the advantage of PDS learning over other algorithms still persists, with a performance close to the optimum achieved by value iteration.

B. Deferrable Loads with Expiration Deadlines

In Section IV, we do not explicitly consider the expiration of deferrable loads. In this section, we discuss how the demand expiration impacts our formulation and the performance of PDS learning. In particular, we assume that a consumer's demand at each time slot expires after K time slots [18].[11] The MDP is revised accordingly as follows:

- The *state* is defined as a tuple $s_{dl}^{(t)} \triangleq (q^{(t)}, b^{(t)}, e^{(t)}, y_1^{(t)}, \cdots, y_{K-1}^{(t)})$. Here $y_i^{(t)}$ represents the amount of unfulfilled demand which expires in i time slots.
- The *aciton* is defined as a tuple $a_{dl}^{(t)} \triangleq (a^{(t)}, \theta_1^{(t)}, \cdots, \theta_K^{(t)})$. Here $a^{(t)}$ is still the consumer's electricity purchase; $\theta_1^{(t)}, \cdots, \theta_{K-1}^{(t)}$ are the amounts of electricity that the consumer uses to fulfill $y_1^{(t)}, \cdots, y_{K-1}^{(t)}$, respectively, and $\theta_K^{(t)}$ is the amount of electricity to fulfill $d^{(t)}$.

[11]In this section, we assume a simplified model where the delay deadlines for the load demand from different appliances are the same and solely focus on how the delay deadline impacts the learning performance. We relegate the analysis on the comprehensive model with heterogeneous delay deadlines as important future works.

TABLE VII
COMPARISON WITH OTHER ALGORITHMS WHEN THERE IS RENEWABLE GENERATION DYNAMICS

Per time slot	***Value iteration***	***RTDP***	***Q-learning***	***PDS learning***
Average utility	1.1312	0.7427	0.7639	1.1102
Average consumed energy	0.5726 kWh	0.3982 kWh	0.4020 kWh	0.4938 kWh
Average purchased price	$0.2119/kWh	$0.2817/kWh	$0.2898/kWh	$0.2210/kWh
Per time slot	***Price prediction***	***Load prediction***	***Price & load prediction***	***PDS learning***
Average utility	0.4919	0.4537	0.5643	1.1102
Average consumed energy	0.2019 kWh	0.2204 kWh	0.3065 kWh	0.4938 kWh
Average purchased price	$0.5687/kWh	$0.6324/kWh	$0.3535/kWh	$0.2210/kWh

TABLE VIII
COMPARISON WITH OTHER LEARNING ALGORITHMS ON DEFERRABLE LOADS

Per time slot ($K = 1$)	***Value iteration***	***RTDP***	***Q-learning***	***PDS learning***
Average utility	1.5618	0.9742	1.0426	1.4487
Average consumed energy	1.7622 kWh	0.8320 kWh	0.8319 kWh	1.8319 kWh
Average purchased price	$2031/kWh	$0.3119/kWh	$0.3128/kWh	$0.2501/kWh
Per time slot ($K = 5$)	***Value iteration***	***RTDP***	***Q-learning***	***PDS learning***
Average utility	1.9673	1.2420	1.1098	1.8653
Average consumed energy	1.9425 kWh	0.8635 kWh	0.8827 kWh	1.9310 kWh
Average purchased price	$0.1987/kWh	$0.2834/kWh	$0.3003/kWh	$0.2225/kWh

TABLE IX
COMPARISON WITH OTHER LEARNING ALGORITHMS AT THE EXISTENCE OF CHARGING/DISCHARGING COSTS

Per time slot	***Value iteration***	***RTDP***	***Q-learning***	***PDS learning***
Average utility	0.9847	0.5748	0.4983	0.9533
Average consumed energy	0.5948 kWh	0.3346 kWh	0.2987 kWh	0.5672 kWh
Average purchased price	$0.2122/kWh	$0.3019/kWh	$0.3232/kWh	$0.2312/kWh

- The *state transition probability* is expressed as follows:

$$\begin{aligned}
&p(s_{dl}^{(t+1)}|s_{dl}^{(t)}, a_{dl}^{(t)}) \\
&= p_q(q^{(t+1)}|q^{(t)}, e^{(t)})p_e(e^{(t+1)}|e^{(t)})p_d(d^{(t)}|e^{(t)}) \\
&I\left(b^{(t+1)} = \min\{\max\{b^{(t)} + a^{(t)} - \sum_{k=1}^{K} \theta_k^{(t)}, 0\}, B\}\right) \\
&I\left(y_{K-1}^{(t+1)} = \max\{0, d^{(t)} - \theta_K^{(t)}\}\right) \\
&\prod_{k=1}^{K-2} I\left(y_k^{(t+1)} = \max\{y_{k+1}^{(t)} - \theta_{k+1}^{(0)}, 0\}\right).
\end{aligned} \tag{28}$$

- The consumer's one-slot utility is

$$u(s_{dl}^{(t)}, a_{dl}^{(t)}) = f(\sum_{k=1}^{K} \theta_k^{(t)}) - q^{(t)}a^{(t)} - cb^{(t+1)}. \tag{29}$$

The performance of the learning algorithms under this model is illustrated in Table VIII. As the expiration deadline K increases, the average consumption per time slot increases while the average energy price decreases, this indicates that a larger expiration deadline enables the consumer to better schedule its energy purchase across time and as a result, the average utility per time slot monotonically increases.

C. Cost of Storage Device Charging and Discharging

In smart grid systems, it is also important to consider the storage device charging and discharging costs, since the life of the battery is usually limited. In the last part of this section, we discuss how the charging/discharging cost impacts the consumer's utility. Specifically, we consider the cost of charging one unit electricity to the battery is α and the cost of discharging one unit electricity from the battery is β. By adding these costs, the consumer's one-slot utility can be reformulated as:

$$\begin{aligned}
u(s^{(t)}, a^{(t)}) &= f(r^{(t)}) - q^{(t)}a^{(t)} - cb^{(t+1)} \\
&- \alpha\left(\min\{B - b^{(t)}, \max\{a^{(t)} - d^{(t)}, 0\}\}\right) \\
&- \beta\left(\max\{d^{(t)} - a^{(t)}, 0\}\right).
\end{aligned} \tag{30}$$

With this change, the experimental result with the PDS learning is illustrated in the following table. As expected, the average utility and the average amount of consumption per slot decrease compared to the scenario where there is no charging/discharging costs. Importantly, PDS learning delivers a high performance significantly outperforming RTDP and Q-learning. Therefore, PDS learning is still applicable in the scenario where charging/discharging costs are considered.

VII. Conclusions

In this paper, we propose a price-aware energy storage management algorithm for electricity consumers who possess electric storage devices. Our algorithm can be applied in smart grids where the demand loads are either non-deferrable or deferrable. They are able to learn the optimal storage management policy without requiring any a priori knowledge of the system dynamics. By introducing the post-decision state and batch update, we prove that our proposed algorithms provide significantly faster convergence speed and thus the average utility received by each individual consumer from the electricity consumption are better compared to the state-of-the-art online reinforcement learning algorithms and energy storage management algorithms.

References

[1] M. Alizadeh, X. Li, Z. Wang, A. Scaglione, and R. Melton, “Demand-Side Management in the Smart Grid: Information Processing for the Power Switch,” IEEE Signal Processing Magazine, vol. 29, no. 5, pp. 55 - 67, 2012.

[2] S. Borenstein,“The Long-run Efficiency of Real-time Electricity Pricing,” Energy J., vol. 26, no. 3, pp. 93 - 116, 2005.

[3] K. Nygard, S. Ghosn, M. Chowhury, D. Loegering, R. McCulloch, and P. Ranganathan, “Optimization Models for Energy Reallocation in a Smart Grid,” IEEE INFOCOM Workshops, 2011.

[4] K. Clement-Nyns, E. Haesen, and J. Driesen, “The Impact of Charging Plug-in Hybrid Electric Vehicles on a Residential Distribution Grid,” IEEE Trans. on Power Systems, vol. 25, no. 1, pp. 371 - 380, 2010.

[5] C. Su and D. Kirschen, “Quantifying the Effect of Demand Response on Electricity Markets,” IEEE Trans. on Power Systems, vol. 24, no. 3, pp. 1199 - 1207, 2009.

[6] H. Song, C. Liu, J. Lawarree, and R. Dahlgren, “Optimal Electricity Supply Bidding by Markov Decision Process,” IEEE Trans. on Power Systems, vol. 15, no. 2, pp. 618 - 624, 2000.

[7] A. Lam, L. Huang, A. Silva, and W. Saad, “A Multi-layer Market for Vehicle-to-Grid Energy Trading in the Smart Grid” IEEE INFOCOM Workshop, 2012.

[8] K. Herter, “Residential Implementation of Critical Peak Pricing of Electricity,” Energy Policy, vol. 35, pp. 2121 - 2130, 2007.

[9] M. He, S. Murugesan, and J. Zhang, “Multiple Timescale Dispatch and Scheduling for Stochastic Reliability in Smart Grids with Wind Generation Integration,” IEEE INFOCOM, 2011.

[10] B. Kim, S. Ren, M. van der Schaar, and J. Lee, “Bidirectional Energy Trading for Residential Load Scheduling and Electric Vehicles,” IEEE INFOCOM, 2013.

[11] I. Atzeni, L. Ordonez, G. Scutari, D. Palomar, and J. Fonollosa, “Noncooperative and Cooperative Optimization of Distributed Energy Generation and Storage in the Demand-side of the Smart Grid,” IEEE Trans. on Signal Process., vol. 61, no. 10, pp. 2454 - 2472, 2013.

[12] A. Mohsenian-Rad and A. Leon-Garcia, “Optimal Residential Load Control with Price Prediction in Real-time Electricity Pricing Environments,” IEEE Trans. on Smart Grid, vol. 1, no. 2, pp. 120 - 133, 2010.

[13] D. Bunn, “Forecasting Loads and Prices in Competitive Power Markets,” Proc. IEEE, vol. 88, pp. 163 - 169, 2000.

[14] L. Jiang and S. Low, “Multi-period Optimal Procurement and Demand Responses in the Presence of Uncertain Supply,” IEEE Conf. on Decision and Control, 2011.

[15] L. Jia and L. Tong, “Optimal Pricing for Residential Demand Response: A Stochastic Optimization Approach,” Allerton Conf. on Communications, Control, and Computing, 2012.

[16] I. Koutsopoulos, V. Hatzi, and L. Tasiulas, “Optimal Energy Storage Control Policies for the Smart Power Grid,” IEEE SmartGridComm, 2012.

[17] P. Reddy and M. Veloso, “Strategy Learning for Autonomous Agents in Smart Grid Markets,” Int’l Joint Conf. on Artificial Intelligence, 2011.

[18] M. Alizadeh, A. Scaglione, and R. Thomas, “From Packet to Power Switching: Digital Direct Load Scheduling,” IEEE J. on Sel. Areas in Commun., vol. 30, no. 6, pp. 1027 - 1036, 2012.

[19] L. Huang, J. Walrandy, and K. Ramchandran, “Optimal Demand Response with Energy Storage Management,” IEEE Conf. on Smart Grid Commun., 2012.

[20] P. Vytelingum, T. Voice, S. Ramchurn, A. Rogers, and N. Jennings, “Agent-based Micro-storage Management for the Smart Grid,” Int’l Conf. on Autonomous Agents and Multiagent Systems (AAMAS), 2010.

[21] A. Mohsenian-Rad, V. 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. on Smart Grid, vol. 1, no. 3, pp. 320 - 331, 2010.

[22] “PowerCentsDC Program Final Report,” Available Online, http://energy.gov/sites/prod/files/oeprod/DocumentsandMedia/DC_OPC_Attachment.pdf.

[23] S. Mirasgedis, Y. Sarafidis, E. Georgopoulous, D. Lalas, M. Moschovits, F. Karagiannis, and D. Papakonstantinous, “Models for mid-term electricity demand forecasting incorporating weather influences,” Energy, vol. 31, no. 2, pp. 208 - 227, 2006.

[24] M. Erol-Kantarci and H. Mouftah, “Wireless Sensor Networks for Cost-Efficient Residential Energy Management in the Smart Grid,” IEEE Trans. on Smart Grid, vol. 2, no. 2, pp. 314 - 325, 2011.

[25] R. Sutton and A. Barto, “Reinforcement Learning: An Introduction,” MIT Press, 1998.

[26] S. Deilami, A. Masoum, P. Moses, and M. Masoum, “Real-Time Coordination of Plug-In Electric Vehicle Charging in Smart Grids to Minimize Power Losses and Improve Voltage Profile,” IEEE Trans. on Smart Grid, vol. 2, no. 3, pp. 456 - 467, 2011.

[27] S. Ren, J. Park, and M. van der Schaar, “Entry and Spectrum Sharing Scheme Selection in Femtocell Communications Markets,” IEEE Trans. on Networking, vol. 21, no. 1, pp. 218 - 232, 2012.

[28] V. Borkar and S. Meyn “The ODE Method for Convergence of Stochastic Approximation and Reinforcement learning,” SIAM J. Control Optimization, vol. 28, pp. 447 - 469, 1999.

[29] Y. Zhang, F. Fu, and M. van der Schaar, “On-Line Learning and Optimization for Wireless Video Transmission,” IEEE Trans. on Signal Process., vol. 58, no. 6, pp. 3108 - 3124, 2010.

[30] D. Bertsekas, “Dynamic Programming and Optimal Control,” Athena Scientific, 2005.

[31] P. Harsha and M. Dahleh, “Optimal Sizing of Energy Storage for Efficient Integration of Renewable Energy,” IEEE Conf. on Decision and Control, pp. 5813 - 5819, 2011.