Efficient Algorithms for Renewable Energy Allocation to Delay Tolerant Consumers

Michael J. Neely, Arash Saber Tehrani, Alexandros G. Dimakis

***Abstract*— We investigate the problem of allocating energy from renewable sources to flexible consumers in electricity markets. We assume there is a renewable energy supplier that provides energy according to a time-varying (and possibly unpredictable) supply process. The plant must serve consumers within a specified delay window, and incurs a cost of drawing energy from other (possibly non-renewable) sources if its own supply is not sufficient to meet the deadlines. We formulate two stochastic optimization problems: The first seeks to minimize the time average cost of using the other sources (and hence strives for the most efficient utilization of the renewable source). The second allows the renewable source to dynamically set a price for its service, and seeks to maximize the resulting time average profit. These problems are solved via the Lyapunov optimization technique. Our resulting algorithms do not require knowledge of the statistics of the time-varying supply and demand processes and are robust to arbitrary sample path variations.**

I. Introduction

The highly variable and unpredictable nature of some renewable energy sources (such as wind and solar) has been a major obstacle to their integration. For example, a recent study conducted by Enernex for wind power integration in Minnesota [5] indicates that the variability and day-ahead forecast errors will result in an additional $\$2.11 - \$4.41$ (for 15% and 25% penetration) per MWh of delivered wind power. Along the same lines, the CAISO report [6] predicted that ten minute real-time energy prices could increase substantially due to wind forecasting errors and identified day-ahead and same-day forecasts and modeling as important tasks for integration of renewable resources.

The necessity to offset variability by stand-by generators and system backup investments substantially increases the cost of renewables. One approach that can mitigate this problem is to couple this supply variability to demand side flexibility [2], [3], [4]. The renewable power suppliers could sell their energy at a lower price to consumers that are willing to wait in a queue, given that it will be served to them within a pre-agreed deadline. This essentially allows a lower price of renewable energy to consumers willing to provide this extra time flexibility. The renewable power supplier can now use this flexibility to deliver the energy when it is available.[1] The supplier will sometimes, hopefully rarely, be in a situation when a prior deadline commitment cannot be matched and will have to purchase the extra energy from the energy spot market (or maintain a costly system backup). Papavasiliou and Oren [4] introduced this problem and proposed an exact backward dynamic programming algorithm and an efficient approximate dynamic programming algorithm for the scheduling decisions of the renewable energy supplier.

In this paper we build a similar model and utilize the technique of Lyapunov optimization initially developed in [11][12][13] for dynamic control of queueing systems for wireless networks. We show that the queuing model naturally fits in the renewable supplier scheduling problem and present a simple energy allocation algorithm that does not require prior statistical information and is provably close to optimal. The proposed framework can be extended to include pricing, multiple queues (with different deadlines) and different objective functions, building on the general results from [11]. We finally evaluate the proposed algorithm on actual CAISO spot market and wind energy production data and show substantial reduction to the operating costs for the renewable supplier compared to a simple greedy algorithm.

In particular, we consider a single renewable energy plant that operates in discrete time with unit timeslots $t \in \{0, 1, 2, \ldots\}$, and provides $s(t)$ units of energy on each slot t. The $s(t)$ process corresponds to the renewable supply and is assumed to be time varying and unpredictable. Since we assume no storage, the energy $s(t)$ must either be used or wasted. Demands for this energy arrive randomly according to a process $a(t)$ (being the amount of energy that is requested on slot t). We assume that consumers requesting energy are flexible, and can tolerate their energy requests being satisfied with some delay. The requests are thus stored in a queue. Every slot t, we use all of our supply $s(t)$ to serve the requests in the queue in a First-In-First-Out (FIFO) manner. However, this may not be enough to meet all of the requests within a timely manner, and hence we also decide to purchase an amount of energy $x(t)$ from an outside (possibly non-renewable) plant. Letting $Q(t)$ represent the total energy requests in our queue on slot t, we have the following update equation:

$$Q(t+1) = \max[Q(t) - s(t) - x(t), 0] + a(t) \qquad (1)$$

The value $x(t)$ is a control decision variable, and incurs a cost $x(t)\gamma(t)$ on slot t, where $\gamma(t)$ is a process that specifies the per-unit-cost of using the outside energy supply on slot t. The value of $\gamma(t)$ can represent a current market price for guaranteed energy services from (possibly non-renewable) sources. As such, the decision to use $x(t)$ units of energy on slot t means the outside source agrees to provide this much energy at time $t+K$ for some fixed (and small) integer $K \geq 0$, for the price $x(t)\gamma(t)$. Without loss of generality, we assume throughout that $K = 0$, so that the energy request is removed from our queue on the same slot in which we decide to use

The authors are with the Electrical Engineering department at the University of Southern California, Los Angeles, CA.

This material is supported in part by one or more of the following: the DARPA IT-MANET program grant W911NF-07-0028, the NSF Career grant CCF-0747525.

[1]Note that in this paper we assume no energy storage although it can be naturally incorporated into our framework.

the outside source. In the actual implementation, requests that are served from the outside source can be removed from the primary queue $Q(t)$ but must still wait an additional K slots.

We first look at the problem of choosing $x(t)$ to stabilize our queue $Q(t)$ while minimizing the time average of the cost $x(t)\gamma(t)$ and also providing a guarantee on the maximum delay D_{max} spent in the queue. If the future values of supply, demand, and market price values $(s(t), a(t), \gamma(t))$ were known in advance, one could in principle make $x(t)$ decisions that minimize total time average cost, possibly choosing $x(t) = 0$ for all t if it is possible to meet all demands using only the renewable energy $s(t)$. The challenge is to provide an efficient algorithm without knowing the future. To this end, we first assume the vector process $(s(t), a(t), \gamma(t))$ is independent and identically distributed (i.i.d.) over slots but has an unknown probability distribution. Under this assumption, we develop an algorithm, parameterized by a positive value V, that comes within $O(1/V)$ of the minimum time average cost required to stabilize the queue, with a worst-case delay guarantee that is $O(V)$. The parameter V can be tuned as desired to provide average cost arbitrarily close to optimal, with a tradeoff in delay. We further show that the same algorithm is provably robust to non-i.i.d. situations, and operates efficiently even for arbitrary sample paths for $(s(t), a(t), \gamma(t))$. Finally, we extend the problem to consider pricing decisions at the renewable energy source, so that the requests $a(t)$ are now influenced by the current prices. In this case, we design a related algorithm that maximizes time average profit.

The Lyapunov optimization technique we use [11][12][13] is related to the primal-dual and fluid-model techniques in [14][15][16][17]. The work in [11][12][13] establishes a general $[O(1/V), O(V)]$ performance-congestion tradeoff for stochastic network optimization problems with i.i.d. (and more general ergodic) processes. Recent work in [18][19][21] provides similar results on a sample path basis, without any probabilistic assumptions. We apply these results in our current paper. Further, we extend the theory by introducing a novel virtual queue that turns an average delay constraint of $O(V)$ (which is achievable with the prior analytical techniques) into a *worst case delay guarantee* that is also $O(V)$.

It is useful to distinguish the proposed Lyapunov optimization method that we use in this paper from dynamic programming techniques. Dynamic programming can be used to solve stronger versions of our problem (such as minimizing average cost subject to a delay constraint) see e.g. [4]. However, dynamic programming requires more stringent system modeling assumptions, has a more complex solution that typically requires knowledge of the supply, demand, and market price probabilities, and cannot necessarily adapt if these probabilities change and/or if there are unmodeled correlations in the actual processes. It involves computation of a value function that can be difficult when the state space of the system is large, and suffers from a *curse of dimensionality* when applied to large dimensional systems (such as systems with many queues).

In contrast, Lyapunov optimization is relatively simple to implement, does not need a-priori statistical knowledge, and is robust to non-i.i.d. and non-ergodic behavior. Further, it has no curse of dimensionality and hence can be applied just as easily in extended formulations that have multiple queues corresponding to multiple customers requesting different deadlines, contrary to dynamic programming [4] which would require exponential complexity in the number of users.

The reason for this efficiency is that Lyapunov optimization relaxes the question that dynamic programming asks: Rather than minimizing time average cost subject to a delay constraint, it seeks to push time average cost towards the more ambitious minimum over all possible algorithms that can stabilize the queue (without regard to the delay constraint). It then specifies an explicit bound on the resulting queue congestion, which depends on the desired proximity to the minimum cost (as defined by the $[O(1/V), O(V)]$ performance-congestion tradeoff). However, the resulting time average queue congestion (and delay) that is achieved is not necessarily the optimal that could be achieved over all possible algorithms that yield the same time average performance cost.

In the next section, we formulate the basic model under the assumption that the $(s(t), a(t), \gamma(t))$ vector is i.i.d. over slots, and present the main allocation algorithm. Section III extends to the case when the renewable power source can set a price for its services. These algorithms are provably robust to non-i.i.d. situations and arbitrary sample paths of events, as shown in Section IV-A. Section IV-B presents an experimental evaluation of our algorithm on a real six-month data set and shows substantial gains over a simple greedy scheduling algorithm.

II. The Dynamic Allocation Algorithm

Suppose that the supply process $s(t)$, the request process $a(t)$, and the market price process $\gamma(t)$, as described in the introduction, form a vector $(s(t), a(t), \gamma(t))$ that is i.i.d. over slots with some unknown probability distribution. We further assume the values of $s(t)$, $a(t)$, $\gamma(t)$ are deterministically bounded by finite constants s_{max}, a_{max}, γ_{max}, so that:

$$0 \leq s(t) \leq s_{max}\ , 0 \leq a(t) \leq a_{max}\ , 0 \leq \gamma(t) \leq \gamma_{max}\ \forall t \quad (2)$$

The queue backlog $Q(t)$ evolves according to (1). The decision variable $x(t)$ is chosen every slot t in reaction to the current $(s(t), a(t), \gamma(t))$ (and possibly additional queue state information) subject to the constraint $0 \leq x(t) \leq x_{max}$ for all t, where x_{max} is a finite upper bound. We assume that $x_{max} \geq a_{max}$ so that it is always possible to stabilize the queue $Q(t)$ (and this can be done with one slot delay if we choose $x(t) = x_{max}$ for all t). Define $\overline{c}$ as the time average cost incurred by our control policy (assuming temporarily that our policy yields such a well defined limit):

$$\overline{c} \triangleq \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\left\{\gamma(t)x(t)\right\}$$

We want to find an allocation algorithm that chooses $x(t)$ over time to solve:

$$\text{Minimize:} \quad \overline{c} \quad (3)$$

$$\text{Subject to:} \quad 1) \quad \overline{Q} < \infty \quad (4)$$

$$2) \quad 0 \leq x(t) \leq x_{max}\ \forall t \quad (5)$$

where $\overline{Q}$ is the time average expected queue backlog, defined:

$$\overline{Q} \triangleq \limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{Q(\tau)\}$$

Define c^* as the infimum time average cost associated with the above problem, considering all possible ways of choosing $x(t)$ over time. The value of c^* is an ambitious target because the above problem is defined only in terms of a queue stability constraint and does not impose any additional delay constraint. We shall construct a solution, parameterized by a constant $V > 0$, that satisfies the constraints of the above problem and pushes the average cost within $O(1/V)$ of the optimal value c^*. Further, we show that our algorithm has the additional property that worst case delay is no more than $O(V)$.

A. The Delay-Aware Virtual Queue

We solve the above problem while also maintaining finite worst case delay using the following novel "virtual queue" $Z(t)$: Fix a parameter $\epsilon > 0$, to be specified later. Define $Z(0) = 0$, and define the virtual queue $Z(t)$ for $t \in \{0, 1, 2, \ldots\}$ according to the following update:

$$Z(t+1) = \max[Z(t) - s(t) - x(t) + \epsilon 1_{\{Q(t)>0\}}, 0] \quad (6)$$

where $1_{\{Q(t)>0\}}$ is an indicator function that is 1 if $Q(t) > 0$, and zero else. The intuition is that $Z(t)$ has the same service process as $Q(t)$ (being $s(t) + x(t)$), but now has an arrival process that adds ϵ whenever the actual queue backlog is non-empty. This ensures that $Z(t)$ grows if there are requests in the $Q(t)$ queue that have not been serviced for a long time. If we can control the system to ensure that the queues $Q(t)$ and $Z(t)$ have finite upper bounds, then we can ensure all requests are served with a worst case delay given in the following lemma.[2]

Lemma 1: (Worst Case Delay) Suppose the system is controlled so that $Z(t) \le Z_{max}$ and $Q(t) \le Q_{max}$ for all t, for some positive constants Z_{max} and Q_{max}. Then all requests are fulfilled with a maximum delay of D_{max} slots, where:

$$D_{max} \triangleq \lceil (Q_{max} + Z_{max})/\epsilon \rceil \quad (7)$$

Proof: Consider any slot t for which $a(t) > 0$. We show that the requests $a(t)$ are fulfilled on or before time $t + D_{max}$. Suppose not (we shall reach a contradiction). Then during slots $\tau \in \{t+1, \ldots, t+D_{max}\}$ it must be that $Q(\tau) > 0$ (else the requests $a(t)$ would have been served before slot τ). Thus, $1_{\{Q(\tau)>0\}} = 1$, and from (6) we have that for all $\tau \in \{t+1, \ldots, t+D_{max}\}$:

$$Z(\tau+1) \ge Z(\tau) - s(\tau) - x(\tau) + \epsilon$$

Summing the above over $\tau \in \{t+1, \ldots, t+D_{max}\}$ yields:

$$Z(t+D_{max}+1) - Z(t+1) \ge -\sum_{\tau=t+1}^{t+D_{max}} [s(\tau)+x(\tau)] + D_{max}\epsilon$$

Rearranging and using the fact that $Z(t+1) \ge 0$ and $Z(t + D_{max} + 1) \le Z_{max}$ yields:

$$\sum_{\tau=t+1}^{t+D_{max}} [s(\tau) + x(\tau)] \ge D_{max}\epsilon - Z_{max} \quad (8)$$

[2]In the case when requests are served by the outside source with an additional delay $K > 0$, then this bound is modified in the actual implementation to $\lceil (Q_{max} + Z_{max})/\epsilon \rceil + K$.

Now note that the requests $a(t)$ are first available for service at time $t+1$, and are part of the backlog $Q(t+1)$ (see (1)). Because $Q(t+1) \le Q_{max}$ and because service is FIFO, these requests $a(t)$ are served on or before time $t + D_{max}$ whenever there are at least Q_{max} units of energy served during the interval $\tau \in \{t+1, \ldots, t+D_{max}\}$. Because we have assumed the requests $a(t)$ are *not* served by time $t + D_{max}$, it must be that $\sum_{\tau=t+1}^{t+D_{max}} [s(\tau) + x(\tau)] < Q_{max}$. Using this in (8) yields:

$$Q_{max} > D_{max}\epsilon - Z_{max}$$

This implies that $D_{max} < (Q_{max} + Z_{max})/\epsilon$, contradicting the definition of D_{max} in (7). □

B. Lyapunov Optimization

Define $\mathbf{\Theta}(t) \triangleq (Z(t), Q(t))$ as the concatenated vector of the real and virtual queues. As a scalar measure of the congestion in both the $Z(t)$ and $Q(t)$ queues, we define the following Lyapunov function: $L(\mathbf{\Theta}(t)) \triangleq \frac{1}{2}[Z(t)^2 + Q(t)^2]$. Define the conditional 1-slot Lyapunov drift as follows:

$$\Delta(\mathbf{\Theta}(t)) \triangleq \mathbb{E}\{L(\mathbf{\Theta}(t+1)) - L(\mathbf{\Theta}(t)) | \mathbf{\Theta}(t)\} \quad (9)$$

Following the drift-plus-penalty framework of [11][12][13], our control algorithm is designed to observe the current queue states $Z(t)$, $Q(t)$ and the current $(s(t), a(t), \gamma(t))$ vector, and to make a decision $x(t)$ (where $0 \le x(t) \le x_{max}$) to minimize a bound on the following expression every slot t:

$$\Delta(\mathbf{\Theta}(t)) + V\mathbb{E}\{\gamma(t)x(t)|\mathbf{\Theta}(t)\}$$

where V is a positive parameter that will be useful to affect a performance-delay tradeoff. The intuition is that taking actions to minimize $\Delta(\mathbf{\Theta}(t))$ alone would push both queues towards lower backlog but incur a large penatly, and so our approach minimizes a weighted sum of drift and penalty.

Lemma 2: (Drift Bound) For any control policy that satisfies $0 \le x(t) \le x_{max}$ for all t, the drift-plus-penalty expression for all slots t satisfies:

$$\begin{aligned}\Delta(\mathbf{\Theta}(t)) + V\mathbb{E}\{\gamma(t)x(t)|\mathbf{\Theta}(t)\} \le B + V\mathbb{E}\{\gamma(t)x(t)|\mathbf{\Theta}(t)\} \\ + Q(t)\mathbb{E}\{a(t) - s(t) - x(t)|\mathbf{\Theta}(t)\} \\ + Z(t)\mathbb{E}\{\epsilon - s(t) - x(t)|\mathbf{\Theta}(t)\} \quad (10)\end{aligned}$$

where the constant B is defined:

$$B \triangleq \frac{(s_{max} + x_{max})^2 + a_{max}^2}{2} + \frac{\max[\epsilon^2, (s_{max} + x_{max})^2]}{2} \quad (11)$$

Proof: See our technical report [1]. □

C. The Dynamic Algorithm

Minimizing the right-hand-side of the drift-plus-penalty bound (10) every slot t leads to the following dynamic algorithm: Every slot t, observe $Z(t)$, $Q(t)$, $(s(t), a(t), \gamma(t))$, and choose $x(t)$ according to the following optimization:

$$\begin{aligned}\text{Minimize:} \quad & x(t)[V\gamma(t) - Q(t) - Z(t)] \\ \text{Subject to:} \quad & 0 \le x(t) \le x_{max}\end{aligned}$$

Then update the actual and virtual queues $Q(t)$ and $Z(t)$ by (1) and (6). The above minimization for the $x(t)$ decision reduces to the following simple threshold rule:

$$x(t) = \begin{cases} 0 & \text{if } Q(t) + Z(t) \leq V\gamma(t) \\ x_{max} & \text{if } Q(t) + Z(t) > V\gamma(t) \end{cases} \quad (12)$$

The above $x(t)$ value drives the queueing updates (1) and (6). However, note by the $\max[\cdot, 0]$ structure of the $Q(t)$ update in (1) that we may not need to purchase the full $x(t)$ units of energy from the outside plant on slot t. Indeed, define $\tilde{x}(t)$ as the *actual* amount purchased from the plant, given by:

$$\tilde{x}(t) \triangleq \begin{cases} x(t) & \text{if } Q(t) - s(t) \geq x(t) \\ \min[Q(t) - s(t), 0] & \text{otherwise} \end{cases} \quad (13)$$

Then we have $\tilde{x}(t) \leq x(t)$ for all t.

Theorem 1: (Performance Analysis) Suppose $x_{max} \geq \max[a_{max}, \epsilon]$. If $Q(0) = Z(0) = 0$, and if the above dynamic algorithm is implemented with any fixed $\epsilon \geq 0$ and $V > 0$ for all $t \in \{0, 1, 2, \ldots\}$, then:

a) The queues $Q(t)$ and $Z(t)$ are deterministically bounded by Q_{max} and Z_{max} every slot t, where:

$$Q_{max} \triangleq V\gamma_{max} + a_{max} \ , \ Z_{max} \triangleq V\gamma_{max} + \epsilon \quad (14)$$

That is, $Q(t) \leq Q_{max}$ and $Z(t) \leq Z_{max}$ for all t. Thus the constraints (4)-(5) are also satisfied.

b) The worst case delay of any request is:

$$D_{max} = \lceil (2V\gamma_{max} + a_{max} + \epsilon)/\epsilon \rceil \quad (15)$$

c) If the vector $(s(t), a(t), \gamma(t))$ is i.i.d. over slots, and if the ϵ parameter is chosen to satisfy $\epsilon \leq \max[\mathbb{E}\{a(t)\}, \mathbb{E}\{s(t)\}]$, then for all slots $t > 0$ the time average cost satisfies:

$$\frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{\gamma(\tau)\tilde{x}(\tau)\} \leq \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{\gamma(\tau)x(\tau)\} \leq c^* + B/V$$

where B is defined in (11).

This theorem demonstrates the $[O(1/V), O(V)]$ cost-delay tradeoff. To obtain the smallest D_{max}, the ϵ value should be chosen as large as possible while still maintaining $\epsilon \leq \max[\mathbb{E}\{a(t)\}, \mathbb{E}\{s(t)\}]$. We can choose $\epsilon = \mathbb{E}\{a(t)\}$ if this expectation is known. Using $\epsilon = 0$ preserves parts (a) and (c) but does not give a finite D_{max}. More discussion of the $\epsilon = 0$ case is given in Section IV-B.

Proof: (Theorem 1 part (a)) We first show that $Q(t) \leq V\gamma_{max} + a_{max}$ for all t. This is clearly true for $t = 0$ (because $Q(0) = 0$). Suppose it holds for slot t. We show it also holds for slot $t + 1$. Consider the case when $Q(t) \leq V\gamma_{max}$. Then $Q(t+1) \leq V\gamma_{max} + a_{max}$, because the queue can increase by at most a_{max} on any slot (see dynamics (1)). Thus, the result holds in this case.

Now consider the opposite case when $V\gamma_{max} < Q(t) \leq V\gamma_{max} + a_{max}$. In this case, we have:

$$Q(t) + Z(t) \geq Q(t) > V\gamma_{max} \geq V\gamma(t)$$

and hence the algorithm will choose $x(t) = x_{max}$ according to (12). If $Q(t) - x_{max} - s(t) > 0$, then on slot t we serve at least x_{max} units of data. Because arrivals $a(t)$ are at most a_{max} (and $a_{max} \leq x_{max}$), the queue cannot increase on the next slot and so $Q(t + 1) \leq Q(t) \leq V\gamma_{max} + a_{max}$. Finally, if $Q(t) - x_{max} - s(t) \leq 0$, then by (1) we have $Q(t + 1) = a(t) \leq a_{max}$, again being less than or equal to $V\gamma_{max} + a_{max}$.

Therefore, $Q(t) \leq V\gamma_{max} + a_{max}$ for all t. The proof that $Z(t) \leq V\gamma_{max} + \epsilon$ for all t is similar and omitted for brevity. □

Proof: (Theorem 1 part (b)) This follows immediately from Lemma 1 together with part (a). □

The proof of Theorem 1 part (c) uses the Lyapunov optimization technique [11], and is given in [1].

III. Pricing for Maximum Profit

We now extend the problem to consider pricing decisions. Instead of a process $a(t)$ that represents requests arriving at slot t, we define a process $y(t)$, called the *demand state* on slot t. The demand state captures any properties of the demand that may affect requests for the renewable energy source in reaction to the price advertised on slot t. A simple example is when $y(t)$ can take one of two possible values, such as HIGH and LOW, representing different demand conditions (such as during peak times or non-peak times for requesting energy). Another example is when $y(t)$ represents the number of consumers willing to purchase renewable energy on slot t. We assume the demand state $y(t)$ is known at the beginning of each slot t (we show a particular case where $y(t)$ does not need to be known after our algorithm is stated).

Every slot t, in addition to choosing the amount of energy $x(t)$ purchased from outside sources, the renewable energy plant makes a binary decision $b(t) \in \{0, 1\}$, where $b(t) = 1$ represents a willingness to accept new requests on slot t, and $b(t) = 0$ means no requests will be accepted. If $b(t) = 1$ is chosen, the plant also chooses a per-unit-energy price $p(t)$ within an interval $0 \leq p(t) \leq p_{max}$, where p_{max} is a pre-established maximum price. The arriving requests $a(t)$ are then influenced by the current price $p(t)$, the current market price $\gamma(t)$, and the current demand state $y(t)$, according to a general *demand function* $F(p, y, \gamma)$. Specifically, the values of $a(t)$ are assumed to be conditionally i.i.d. over all slots with the same $p(t)$, $y(t)$, $\gamma(t)$, and satisfy:

$$\mathbb{E}\{a(t)|p(t), y(t), \gamma(t), b(t) = 1\} = F(p(t), y(t), \gamma(t))$$

We assume the function $F(p, y, \gamma)$ is continuous in p for each given y and γ.[3] We further assume the arrivals $a(t)$ continue to be worst-case bounded by a_{max}, regardless of $p(t)$, $y(t)$, $\gamma(t)$. The queue iteration $Q(t)$ still operates according to (1), with the understanding that $a(t)$ is now influenced by the pricing decisions. Let $\phi(t)$ represent the instantaneous profit earned on slot t, defined as:

$$\phi(t) = b(t)p(t)a(t) - \gamma(t)x(t)$$

We now consider the following problem:

$$\text{Maximize:} \quad \overline{\phi} \quad (16)$$

$$\text{Subject to:} \quad 1) \quad \overline{Q} < \infty \quad (17)$$

$$2) \quad 0 \leq x(t) \leq x_{max} \forall t \quad (18)$$

$$3) \quad b(t) \in \{0, 1\} \ , \ 0 \leq p(t) \leq p_{max} \forall t \quad (19)$$

[3]This continuity is only used to ensure the resulting min-drift decision has a well defined minimizing price $p(t)$ every slot.

where $\overline{\phi}$ is defined as the limiting time average profit:

$$\overline{\phi} \triangleq \lim_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{\phi(\tau)\}$$

To solve the problem, we use the same queueing structure for $Q(t)$ and $Z(t)$ in (1) and (6), and the same Lyapunov function $L(\mathbf{\Theta}(t))$. However, we now consider the "penalty" $-\phi(t)$, and so the drift-plus-penalty technique seeks to choose a vector that minimizes a bound on:

$$\Delta(\mathbf{\Theta}(t)) - V\mathbb{E}\{\phi(t)|\mathbf{\Theta}(t)\}$$

Using the same analysis as Lemma 2, we can show the following bound on this drift-plus-penalty expression:

$$\begin{aligned}\Delta(\mathbf{\Theta}(t)) - V\mathbb{E}\{\phi(t)|\mathbf{\Theta}(t)\} \leq B\\ -V\mathbb{E}\{b(t)p(t)F(p(t),y(t),\gamma(t)) - \gamma(t)x(t)|\mathbf{\Theta}(t)\}\\ +Q(t)\mathbb{E}\{b(t)F(p(t),y(t),\gamma(t)) - s(t) - x(t)|\mathbf{\Theta}(t)\}\\ +Z(t)\mathbb{E}\{\epsilon - s(t) - x(t)|\mathbf{\Theta}(t)\} \quad (20)\end{aligned}$$

Our joint energy-allocation and pricing algorithm observes the current system state on each slot t, and chooses $b(t)$, $p(t)$, and $x(t)$ to minimize the right-hand side of the above drift expression (given the observed $\mathbf{\Theta}(t)$). This reduces to the following: Every slot t, observe queues $Q(t)$, $Z(t)$, and observe $s(t)$, $\gamma(t)$, $y(t)$. Then choose a price $p(t)$ and an allocation $x(t)$ as follows:

- (Pricing $p(t)$) Choose $p(t)$ as the solution to:

$$\begin{aligned}\text{Max:} \quad & F(p(t),y(t),\gamma(t))(Vp(t) - Q(t))\\ \text{S.t.:} \quad & 0 \leq p(t) \leq p_{max}\end{aligned}$$

  If the resulting maximum value is non-negative, choose $b(t) = 1$. Else choose $b(t) = 0$ so that no new requests are allowed on slot t.
- (Allocating $x(t)$) Choose $x(t)$ according to (12).
- (Queue Updates) Update $Q(t)$ and $Z(t)$ by (1) and (6).

This pricing pricing policy does not need to know the demand state $y(t)$ in the special case when $F(p(t),y(t),\gamma(t)) = y(t)\hat{F}(p(t),\gamma(t))$, so that demand state simply scales the demand function. This pricing structure is similar to that considered in [20] for wireless service providers.

Theorem 2: Assume that $x_{max} \geq \max[a_{max},\epsilon]$, and that $Q(0) = Z(0) = 0$. If the above joint pricing and allocation policy is implemented every slot with fixed parameters $\epsilon \geq 0$, $V > 0$, then:

a) The worst case delay D_{max} and backlog Q_{max} are the same as before (given in (15), (14)), where Q_{max} is proportional to V and D_{max} is proportional to V/ϵ.

b) If the vector $(s(t),y(t),\gamma(t))$ is i.i.d. over slots, and if $\epsilon \leq \mathbb{E}\{s(t)\}$, then:[4]

$$\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{\phi(\tau)\} \geq \phi^* - B/V \quad \forall t > 0$$

where B is defined in (11), and ϕ^* is the optimal time average profit that can be achieved by any algorithm that satisfies the constraints of the problem (16)-(19).

Proof: See [1]. □

[4]Note that *actual profit* can be defined $\tilde{\phi}(t) \triangleq b(t)p(t)a(t) - \gamma(t)\tilde{x}(t)$, with $\tilde{x}(t)$ defined in (13). Clearly $\tilde{\phi}(t) \geq \phi(t)$ for all t, and so the time average of the actual profit $\tilde{\phi}(t)$ is even closer to the optimal value ϕ^*.

IV. Experimental Evaluation

A. Robustness to Non-i.i.d. Behavior

We first note that the deterministic bounds Q_{max}, Z_{max}, D_{max} in Theorems 1 and 2 hold for all sample paths that satisfy (2), not just those with i.i.d. properties. This is because our proof used a *sample path argument*. Further, using the *universal scheduling* technique of [18][19][21], we can show that for any (possibly non-i.i.d., non-ergodic) sample path, the same algorithms always yield performance within BT/V of the performance of a "genie aided" policy that makes decisions based on knowledge up to T slots into the future, for any integer $T > 0$. This gap BT/V can be made arbitrarily small if we fix T and choose a large value of V, which also increases delay as $O(V)$. This analysis is omitted for brevity (see [1]).

B. Simulations on Real Data Sets

We evaluated the performance of the proposed algorithm on a six-month data set that we created by combining 10-minute average spot market prices $\gamma(t)$ for Los Angeles area (LA1) from CAISO [22] and 10-minute energy production $s(t)$ for a small subset of windfarms from the Western Wind resources Dataset published by the National Renewable Energy Laboratory [23]. We modeled the demand $a(t)$ as i.i.d. over slots and uniformly distributed over the integers $\{0,1,\ldots,a_{max}\}$. We executed the algorithm in 10-minute timeslots and experimented with different values of the parameters V,ϵ and the corresponding deadlines they generate. Here we present a subset of those results (see [1] for more).

We compare the proposed algorithm against a simple greedy strategy "Purchase at deadline," which tries to use all the available resource $s(t)$ and only buys from the spot market as a last resort if a deadline is reached. As can be seen in Fig. 1, the proposed algorithm reduces the cost of the renewable supplier by approximately a factor of 2 in the tested six-month window. This is not surprising since the greedy strategy does not hedge for future high prices in the spot market while the proposed algorithm learns to proactively buy when the spot market prices are lower than typical and deadline violations seem probable. The high variability of the spot market prices [22] makes this advantage significant. The second observation, seen in Fig. 2, is that the proposed algorithm has on average a much smaller delay than the deadline, which for our parameters was $D_{max} = 69.1$ hours. On the contrary, the greedy algorithm makes many requests wait close to (or exactly at) the maximum allowed 69.1 hours. Note that 69.1 hours corresponds to 415 slots, and our algorithm maintains delays within 10 hours (60 slots), as shown in Fig. 2. We have used a slot size of 10 minutes because that was the granularity of the available data. If system updates are available every minute or less, then the slot size can be reduced by at least an order of magnitude.

Our results use $\epsilon = \mathbb{E}\{a(t)\} = a_{max}/2$. We also conducted simulations with $\epsilon = 0$, which does not require knowledge of $\mathbb{E}\{a(t)\}$. While $\epsilon = 0$ does not provide a finite delay guarantee, it still guarantees the same finite Q_{max}. Together with FIFO service, this means that the worst case delay for requests that arrive at time t is given by the smallest integer $T > 0$ such that $\sum_{\tau=t+1}^{t+T} s(\tau) \geq Q_{max}$. While

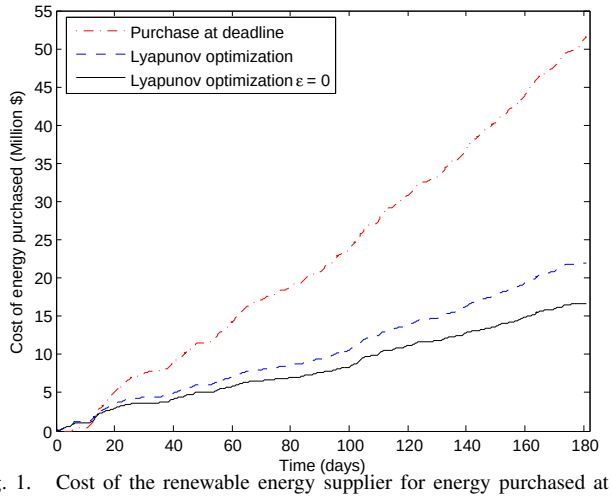

Fig. 1. Cost of the renewable energy supplier for energy purchased at the spot market. For the proposed algorithm we used the parameters $a_{max} = 175, \gamma_{max} = 180, x_{max} = 400, V = 100, D_{max} = 415 = 2.9$ days.

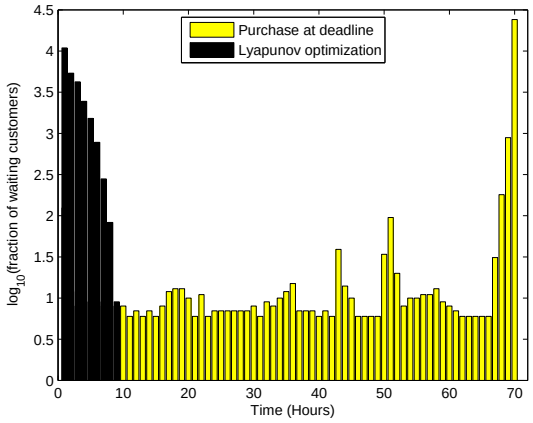

Fig. 2. Histogram of delay for the customers waiting in the service queues of the renewable energy supplier under the two algorithms (vertical axis in logarithmic scale). Case $\epsilon = 0$ is not shown, but has max delay 14 hours, as compared with the $\epsilon = \mathbb{E}\{A(t)\}$ case (shown) with max delay 9.5 hours.

there is no bound on this for general $s(t)$ processes, it can still lead to small delays. Indeed, in the simulations it *still* maintained all delays under $D_{max} = 2.9$ days (having a maximum experimental delay of 14 hours, as compared to 9.5 hours for the $\epsilon = \mathbb{E}\{a(t)\}$ case).[5] Fig. 1 shows it gives slightly better cost, particularly because it increases delay. Both Lyapunov optimization algorithms provided significantly better cost and delay as compared to the greedy algorithm. It should be noted that we did not compare against dynamic programming algorithms such as the one proposed in [4]. While it is clear that a dynamic programming approach could solve this problem optimally if the statistics of the underlying processes were known, one benefit of our approach is that no such prior knowledge is required. Further, while we treat only the 1-queue case here, the Lyapunov approach can be used to provide an efficient algorithm for multiple queues corresponding to different customers with different deadlines.

REFERENCES

[1] M. J. Neely, A. Saber Tehrani, and A. G. Dimakis, Efficient Algorithms for Renewable Energy Allocation to Delay Tolerant Consumers. *ArXiv Technical Report, arXiv:1006.4649*, June 2010.

[2] A. Papavasiliou and S. S. Oren, Coupling Wind Generation with Deferrable Loads. *Proceedings of the IEEE Energy 2030 Conference*, Atlanta, Georgia November 17-18, 2008.

[3] A. Papavasiliou, S. S. Oren, M. Junca, A.G. Dimakis, and T. Dickhoff, Coupling Wind Generators with Deferrable Loads. *CITRIS White paper Technical Report*, 2008.

[4] A. Papavasiliou and S. S. Oren, Supplying Renewable Energy to Deferrable Loads: Algorithms and Economic Analysis. *IEEE PES General Meeting*, Minneapolis, Minnesota, July 25-29 2010.

[5] R. Zavadil, "2006 Minessota Wind Integration Study," in *The Minnesota Public Utilities Commission Technical report, Vol I*, Nov 2006.

[6] C. Loutan and D. Hawkins, "Integration of renewable resources. Transmission and operating issues and recommendations for integrating renewable resources on the California ISO-controlled Grid," in *Technical report, California Independent System Operator*, 2007.

[7] R. Sioshansi and W. Short, "Evaluating the impacts of real-time pricing on the usage of wind power generation," in *The Economics of Energy Markets*, June 2008.

[8] `http://apps1.eere.energy.gov/news/news_detail.cfm/news_id=12230`.

[9] `http://www.doe.energy.gov/smartgrid.htm`.

[10] F. van Hulle. Large scale integration of wind energy in the european power supply: Analysis, recommendations and issues. *Technical Report, European Wind Energy Association*, 2005.

[11] L. Georgiadis, M. J. Neely, and L. Tassiulas. Resource allocation and cross-layer control in wireless networks. *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1-149, 2006.

[12] M. J. Neely. Energy optimal control for time varying wireless networks. *IEEE Transactions on Information Theory*, vol. 52, no. 7, pp. 2915-2934, July 2006.

[13] M. J. Neely. *Dynamic Power Allocation and Routing for Satellite and Wireless Networks with Time Varying Channels*. PhD thesis, Massachusetts Institute of Technology, LIDS, 2003.

[14] A. Eryilmaz and R. Srikant. Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control. *Proc. IEEE INFOCOM*, March 2005.

[15] A. Stolyar. Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm. *Queueing Systems*, vol. 50, pp. 401-457, 2005.

[16] R. Agrawal and V. Subramanian. Optimality of certain channel aware scheduling policies. *Proc. 40th Annual Allerton Conference on Communication , Control, and Computing, Monticello, IL*, Oct. 2002.

[17] H. Kushner and P. Whiting. Asymptotic properties of proportional-fair sharing algorithms. *Proc. of 40th Annual Allerton Conf. on Communication, Control, and Computing*, 2002.

[18] M. J. Neely. Universal scheduling for networks with arbitrary traffic, channels, and mobility. *ArXiv technical report*, arXiv:1001.0960v1, Jan. 2010.

[19] M. J. Neely and L. Huang. Dynamic product assembly and inventory control for maximum profit. *ArXiv Technical Report*, April 2010.

[20] L. Huang and M. J. Neely. The Optimality of Two Prices: Maximizing Revenue in a Stochastic Communication System. *IEEE/ACM Transactions on Networking*, vol. 18, no. 2, pp. 406-419, April 2010.

[21] M. J. Neely. Stock Market Trading via Stochastic Network Optimization. ArXiv Technical Report, arXiv:0909.3891v1, Sept. 2009.

[22] California ISO Open Access Same-time Information System (OASIS) 10-Minute Settlement Interval Average Prices `http://oasishis.caiso.com/`

[23] Western Wind resources Dataset, The National Renewable Energy Laboratory `http://wind.nrel.gov/Web_nrel/`

[5] For legibility, the delay data for the $\epsilon = 0$ case is not shown in Fig. 2.