Optimal Sleep-Wake Scheduling for Energy Harvesting Smart Mobile Devices

Longbo Huang
longbohuang@tsinghua.edu.cn
IIIS, Tsinghua University

***Abstract*—In this paper, we develop optimal sleep/wake scheduling algorithms for smart mobile devices that are powered by batteries and are capable of harvesting energy from the environment. Using a novel combination of the two-timescale Lyapunov optimization approach and weight perturbation, we first design the Optimal Sleep/wake scheduling Algorithm (OSA), which *does not* require any knowledge of the harvestable energy process. We prove that OSA is able to achieve any system performance that is within $O(\epsilon)$ of the optimal, and explicitly compute the required battery size, which is $O(1/\epsilon)$. We then extend our results to incorporate system information into algorithm design. Specifically, we develop the Information-aided OSA algorithm (IOSA) by introducing a novel drift augmenting idea in Lyapunov optimization. We show that IOSA is able to achieve the $O(\epsilon)$ close-to-optimal utility performance and ensures that the required traffic buffer and energy storage size are $O(\log(1/\epsilon)^2)$ with high probability.**

I. Introduction

According to a recent technology report [1], the number of smart mobile devices is increasing very rapidly and will soon exceed the world population. With the rapidly increasing computing power, these mobile smart devices will become a very important component of our daily computing resource. However, the limited battery life has been one of the most constrained resources of these new powerful devices [2]. To resolve this problem, efforts have been made to enable the devices to "harvest" energy from the environment. For instance, by converting ambient radio power into energy [3], by converting mechanical vibration into energy [4], or by using solar panels [5]. These new energy harvesting technologies can likely be a remedy for the poor battery performance of smart devices and greatly improve user experience.

However, to take full advantage of the energy harvesting capability, efficient energy management algorithms for such smart mobile devices must be developed. In this paper, we consider the problem of constructing utility optimal sleep/wake scheduling algorithms for a single smart mobile device system. The system operates in frames which consist of multiple time slots. In every frame, the device may receive external requests for performing computing tasks, e.g., from the device user or software applications. Besides fulfilling such computing demand, the system also supports a set of data flows for applications. Thus, in every frame, the device first decides whether to enter the sleep mode or to stay awake during the frame. If it stays awake, in every time slot of the frame, it determines how much computing demand to fulfill, how much traffic to admit for the flows it supports, and how much power to spend for packet delivery. If instead the node enters the sleep mode, it turns off the transmission module and does not respond to the external requests. The system receives utility by delivering data but may suffer from disutility due to partial fulfillment of the computing demand. The objective of the node is to maximize the aggregate flow utility minus disutility, subject to the constraint that the average data backlog is finite, and that the "energy-availability" constraint is satisfied at all time, i.e., the energy consumed is no more than the energy stored. This "energy-availability" constraint greatly complicates the design of an efficient scheduling algorithm, as the current energy expenditure decision can cause energy outage in the future and affect future decisions. Problems involving such no-underflow constraints can in principle be formulated as dynamic programs (DP) and solved optimally. However, the DP approach requires substantial statistical knowledge of the system dynamics, and often runs into the "curse-of-dimensionality" problem when the state space is large.

There have been many previous works developing algorithms for such energy harvesting systems. [6] develops algorithms for a single sensor node for achieving maximum capacity and minimum delay when the rate-power curve is linear. [7] looks at the problem of designing energy-efficient schemes for maximizing the decay exponent of the queue length. [8] develops scheduling algorithms to achieve close-to-optimal utility for energy harvesting networks with time varying channels. [9] develops an energy-aware routing scheme that approaches optimal as the network size increases. [10] designs optimal control schemes for general multihop energy harvesting networks. [11] considers a similar scenario and handles compression. [12] constructs optimal sleep/wake schemes for a single node system. [13] designs transmission policies for energy harvesting sensors with time-correlated energy supply. [14] and [15] apply the reinforcement learning approach for solving the energy management problem for a single sensor node and derive heuristic algorithms. However, most of the aforementioned works assume that the nodes in the system always remain "ON" and design power allocation decisions, whereas in practice, nodes typically go through sleep/wake cycles [16]. Hence, previous results do not consider

This work was presented in part at the 11th International Symposium and Workshops on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt) 2013, Tsukuba Science City, Japan.

This work was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61361136003, 61303195, the China youth 1000-talent grant.

the two-timescale operation pattern of the nodes and are not directly applicable to our problem.

We tackle this problem using two novel approaches. The first approach combines the two-timescale Lyapunov optimization technique developed in [17] and *weight perturbation* developed in [18] and [19]. The idea of this approach is to construct the algorithm based on a multi-slot Lyapunov drift and carefully perturb the weights used for decision making, so as to "push" the target energy level towards certain nonzero values to avoid energy outage. Based on this idea, we develop the Optimal Sleep/wake scheduling Algorithm (OSA) for achieving optimal utility. OSA is an *online* algorithm which makes greedy decisions every frame and *does not* require any knowledge of the harvestable energy process. We show that the OSA algorithm is able to achieve an average utility that is within $O(\epsilon)$ of the optimal for any $\epsilon > 0$, and only requires an energy storage device that is of size $O(1/\epsilon)$. We also explicitly compute the required storage capacity and show that OSA also guarantees that the data traffic congestion in the system is deterministically bounded by $O(1/\epsilon)$. Such an explicit characterization is particularly useful for practical implementations.

Our second approach extends the first one by introducing an augmentation idea into Lyapunov optimization and leads to the Information-aided OSA algorithm (IOSA). IOSA provides an explicit way for incorporating system information into algorithm design and demonstrates the potential benefits for doing so. Different from OSA, IOSA relaxes the deterministic buffer level guarantees but provides strong probabilistic bounds on the energy outage events. We show that IOSA achieves the similar $O(\epsilon)$ near-optimal utility performance while ensuring that the required buffer sizes are $O(\log(1/\epsilon)^2)$ with high probability.

Our paper is mostly related to the recent works [10], [11] and [8], which use a similar Lyapunov optimization approach for algorithm design. However, the problems there do not consider the sleep/wake operation pattern of the nodes in the system. Moreover, the algorithms developed there do not incorporate system information into control. Hence, the problem considered in this paper can be viewed as a generalization of problems studied in [8] and [10] for the single node case, and requires a very different analysis from the typically Lyapunov drift analysis adopted in [10] and [11]. Our work is also very different from earlier works [20] and [21] as follows: (i) [20] and [21] consider a static setting; whereas the system can be highly dynamic here. (ii) the algorithm design and performance analysis process in [20] and [21] relies on leaky-bucket assumptions; whereas in our case, we allow general control policies and explicitly characterize traffic delay. (iii) [20] and [21] assumes that the node is always ON and focus on power allocation; whereas we take into account the fact that nodes can choose to enter the sleep mode or stay awake. Since mobile devices typically consume significant power when it is idle [22], always keeping the phone ON consumes much more power compared to carefully putting it to sleep.

Our paper is organized as follows. In Section II we state our system model and the objective. Section III presents the OSA algorithm design procedure. The $[O(\epsilon), O(1/\epsilon)]$ performance

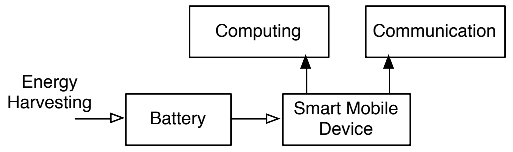

Fig. 1. The system model. The smart mobile device provides computing and communication services to device users or software applications.

results of the OSA algorithm are presented in Section IV. Section V presents the IOSA algorithm and summarizes its performance results. Simulation results are presented in Section VI. We conclude our paper in Section VII.

II. The System Model

We consider a system that consists of a single smart mobile device (called the node in the following), which provides computing and communication services to device users and software applications (Fig. 1). The node is powered by an energy battery and is capable of harvesting energy from the environment. Time is slotted, i.e., $t \in \{0, 1, 2, ...\}$, and is divided in to frames of size T. For notation convenience, we use $\mathbb{T}_m$ to denote the set of slots in frame m, i.e., $\mathbb{T}_m \triangleq [mT, ..., (m+1)T - 1]$.

A. The Demand State and Sleep/Wake Model

In each frame, the node can choose to either stay *awake*, or enter the *sleep* mode. We model the sleep/wake decision by $1_{\text{w}}(m_t)$, where $m_t = \lfloor t/T \rfloor$. That is, $1_{\text{w}}(m_t) = 1$ if the node stays awake during the frame t belongs to. Otherwise $1_{\text{w}}(m_t) = 0$. However, in some frames, the node may receive external commands to perform certain tasks, e.g., a smartphone user wants to perform certain computing task or a software application requires some processing power.[1] To model this situation, we define a *demand* state $\chi(m)$, where $\chi(m) = 1$ means that the node receives external requests to process jobs in frame m, and $\chi(m) = 0$ otherwise.

If $\chi(m) = 1$, in every time slot $t \in \mathbb{T}_m$, the node receives an external power demand $d(t) \in \mathcal{D}$, where $\mathcal{D}$ is the set of possible demands and is assumed to be finite and discrete. We then define $d_{\max} = \max\{d : d \in \mathcal{D}\}$.[2]

After receiving the demand, the node decides how much power to allocate to fulfil the demand. We model this by using $0 \leq b(t) \leq d(t)$ to denote the fulfilled amount. If the node decides to enter the sleep mode in frame m_t, then $b(t) = 0$ for all $t \in \mathbb{T}_{m_t}$.[3] Finally, we use

$$D(t) = D(1_{\text{w}}(m_t)b(t), d(t), \chi(m_t)) \tag{1}$$

to denote the *disutility* incurred due to partial fulfilment of the demand, which measures the "unhappiness" of the entity requesting the power demand. An example of $D(t)$ can be:

$$D(t) = a\chi(m_t)(1_{\text{w}}(m_t)b(t) - d(t))^2, \tag{2}$$

[1]Here we use frames to model the timescale at which users change their usage behavior.

[2]We model all the external tasks purely by the power they consume.

[3]This can be done in the case when external demands are from users trying to use the device. In this case, the node enters the sleep mode after having a short "decision phase" at the beginning of every frame. During the decision phase, the device negotiates with the demand requesting entity, e.g., users, to perform demand response (a concept that has been getting increasing attention and adoption in both the research community and industry, e.g., [23] and [24]) according to the battery level.

where $a > 0$ is a constant. We assume that $d(t) = 0$ and $D(t) = 0$ if $\chi(m) = 0$, i.e., if there is no external demand, then there is no resulting disutility due to partial fulfilment. To characterize the degree of flexibility, we use $\alpha > 0$ to denote the growth rate of the disutility, i.e., for any $0 \leq b_1, b_2 \leq d$, we have:

$$\sup_{d,\chi} |D(b_1, d, \chi) - D(b_2, d, \chi)| \leq \alpha |b_2 - b_1|. \tag{3}$$

That is, the disutility growth rate is no larger than α. Intuitively, a larger α implies that the user is less flexible in partial fulfilment of the workload. Note that such partial fulfilment can be viewed as the node performing demand response [25]. Such scenarios already exist in today's smartphones, where the devices remind the user about the energy level when receiving computing commands.

In the following, we assume for simplicity that $\chi(m)$ is i.i.d. every frame and let $\pi_\chi = \mathbf{Pr}\{\chi(m) = 1\}$. We also assume that conditioning on $\chi(m)$, $d(t)$ is i.i.d. every time slot in a frame and is independent of everything else. We let $\pi_d = \mathbf{Pr}\{d(t) = d\}$ when $\chi(m_t) = 1$.

B. The Traffic Utility Model

Besides satisfying external power demands, the node also provides traffic delivery service to a set of communication flows (called commodities) denoted by $\mathcal{C}$, e.g., file transfer for different software applications.[4] Then, in frames when the node stays awake, the node decides how many commodity $c \in \mathcal{C}$ packets to admit in every time slot. We use $R^{(c)}(t)$ to denote the amount of new commodity c data admitted at time t. We assume that $0 \leq R^{(c)}(t) \leq R_{\max}$ for all c with some finite $R_{\max}$ at all time. Each commodity is associated with a utility function $U^{(c)}(r)$. Each $U^{(c)}(r)$ function is assumed to be increasing, continuously differentiable, and strictly concave in r with a bounded first derivative and $U^{(c)}(0) = 0$. We use β^c to denote the maximum first derivative of $U^{(c)}(r)$, i.e., $\beta^c = (U^{(c)})'(0)$ and denote

$$\beta = \max_c \beta^c. \tag{4}$$

During frames when the node is in the sleep mode, we have $R^{(c)}(t) = 0$ for all time.

C. The Transmission Power Consumption Model

If the node stays awake in a frame, in order to deliver the data to their destinations, the node allocates power for data transmission over a wireless link, e.g., to the base station. To capture the time-varying nature of the wireless link, we denote $S(t)$ the *channel state* of the node, e.g., fading coefficient. We assume that $S(t)$ takes values in some finite set $\mathcal{S} = (s_1, ..., s_{M_s})$, and assume in the following that the pair of energy state (defined later) and $S(t)$ is independent and identically distributed (i.i.d.) every slot. We denote $\pi_s = \mathbf{Pr}\{S(t) = s\}$. At every time slot, if $S(t) = s_i$ and the node stays awake, it chooses a power allocation value $P(t)$ from a feasible power allocation set $\mathcal{P}^{(s_i)}_{\text{awake}}$. We assume that $\mathcal{P}^{(s_i)}_{\text{awake}}$ is compact for all s_i, and that every feasible power allocation in $\mathcal{P}^{(s_i)}_{\text{awake}}$ satisfies the constraint $P_{\text{min}} \leq P(t) \leq P_{\max}$ for some $P_{\text{min}} > 0$ and $P_{\max} < \infty$. Here the P_{min} constraint is to capture the fact that the node will spend a considerable amount of power compared to the sleep mode, even if it simply stays idle and does nothing [22]. On the other hand, if the node decides to enter the sleep mode, then $P(t) = 0$ for all $t \in \mathbb{T}_m$.

Given the channel state $S(t)$ and the power allocation value $P(t)$, the transmission rate is given by the rate-power function $\mu(t) = \mu(S(t), P(t))$. For each s_i, we assume that the function $\mu(s_i, P(t))$ satisfies the following property.

***Property** 1:* For any s_i and any P, we have for some finite constant $\delta > 0$ that:

$$\mu(s_i, P) \leq \delta P. \quad \diamond \tag{5}$$

Property 1 states that the rate obtained over the link is upper bounded by some linear function of the power allocated to it. Such a property can be satisfied by most rate-power functions, e.g., when the rate function is differentiable and has finite directional derivatives with respect to power [26].

From the above, in particular, finite channel states and compact power allocation set, we can see that there exists some finite constant $\mu_{\max}$ such that $\mu(t) \leq \mu_{\max}$ for all time under any $P(t)$ and any channel state $S(t)$. We also assume that $R_{\max} \geq \mu_{\max}$. In the following, we use $\mu^{(c)}(t)$ to denote the rate allocated to the commodity c data at time t. It is easy to see that at any time t, we have:

$$\sum_c \mu^{(c)}(t) \leq \mu(t). \tag{6}$$

D. The Energy Queue Model

The node is assumed to be powered by a battery. We model the battery using an *energy queue* $E(t)$, which measures the amount of the energy stored in the battery at time t. We assume that the node can observe its energy level $E(t)$. In any time slot t, the chosen actions must satisfy the following "energy-availability" constraint for all time:[5]

$$b(t) + P(t) \leq E(t). \tag{7}$$

That is, the consumed power must be no more than what is available.[6]

The node is also assumed to be capable of harvesting energy from the environment, for instance, using solar panels [6]. To model the dynamic nature of the harvestable energy, e.g., due to mobility, we use $h(t)$ to denote the amount of harvestable energy at time t, and call it the *energy state*. We assume that $h(t)$ takes values in some finite set $\mathcal{H} = \{h_1, ..., h_{M_h}\}$ and has an average rate of λ_h. We assume that the pair $[h(t), S(t)]$ is i.i.d. over slots (possibly correlated in the same slot), with distribution $\pi(h, s)$ and marginals π_h, π_s, respectively.[7]

[4] We only consider the case when the other major responsibility of the node is serving data traffic. This traffic model can also be used to model computing workload that are delay tolerant.

[5] This condition assumes that the energy harvested at time t is assumed to be available for use in time $t + 1$. Our results extend easily to allow using the energy harvested in the same slot.

[6] We measure time in unit size slots, so that our power $P(t)$ has units of energy/slot, and $P(t) \times (1\,\text{slot})$ is the resulting energy consumption in one slot.

[7] The energy state is assumed to change every time slot. This is different from the sleep/wake action, which is done every frame. This distinction is introduced to capture the fact that sleep/wake actions are often caused by human users and thus the response timescale is in general longer.

We assume that there exists $h_{\max} < \infty$ such that $0 \leq h(t) \leq h_{\max}$ for all t. In the following, it is convenient for us to assume the energy queue has infinite capacity, and that the node can decide whether or not to harvest energy in each slot. We model this harvesting decision by using $e(t) \in [0, h(t)]$ to denote the amount of energy that is actually harvested at time t. Note here we assume that the energy harvesting action is not affected by the sleep/wake mode of the node. Also note that the introduction of $e(t)$ is important because it enables the development of low-complexity algorithms, which in turn help us understand the key factors in the original energy harvesting network control problem. In actual algorithm implementation, one will run the same algorithm but always harvests energy, i.e., $e(t) = h(t)$. Then, the actual energy level is always no less than that under the proposed algorithms (See Section III for more explanations).

E. Queueing Dynamics

Let $Q(t) = (Q^{(c)}(t), c \in \mathcal{C})$, $t = 0, 1, 2, ...$ be the data queue backlog vector at the node, where $Q^{(c)}(t)$ is the amount of commodity c data buffered. We see that the queue evolves according to:

$$\begin{aligned} &Q^{(c)}(t+1) \qquad (8)\\ &\quad = \left[Q^{(c)}(t) - 1_{\text{w}}(m_t)\mu^{(c)}(t)\right]^+ + 1_{\text{w}}(m_t)R^{(c)}(t), \end{aligned}$$

with $Q^{(c)}(0) = 0$ for all $c \in \mathcal{C}$ and $[x]^+ = \max[x, 0]$. In this paper, we say that the system is *stable* if:

$$\overline{Q} \triangleq \limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \sum_{c} \mathbb{E}\big\{Q^{(c)}(\tau)\big\} < \infty. \qquad (9)$$

Similarly, $E(t)$ denotes the energy queue size. Due to the energy availability constraint (7), we see that the energy queue $E(t)$ evolves according to the following:

$$E(t+1) = E(t) - 1_{\text{w}}(m_t)[b(t) + P(t)] + e(t), \qquad (10)$$

with $E(0) = 0$. [8] By using the queueing dynamic (10), we start by assuming that each energy queue has infinite capacity. We will show later that our first algorithm guarantees a deterministic energy storage bound, while the second algorithm ensures that we only need a small energy storage size with high probability.

F. Utility Maximization with Energy Management

The goal of the system is to design a joint sleep/wake management, flow control, scheduling, and power allocation algorithm, which first chooses the right sleep/wake decision at the beginning of each frame. Then, at every time slot, admits the right amount of data $R^{(c)}(t)$, fulfils the power demand $0 \leq b(t) \leq d(t)$ and chooses power allocation value $P(t)$ subject to (7), and transmits packets accordingly, so as to maximize ϕ, the aggregate flow utility minus the time average disutility, i.e.,

$$\phi \triangleq \sum_{c} \overline{U^{(c)}(t)} - \overline{D(t)}, \qquad (11)$$

subject to the system stability constraint (9). Here $U^{(c)}(t) = U^{(c)}(R^{(c)}(t))$ and the notation $\overline{x(t)} \triangleq \lim_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\big\{x(\tau)\big\}$. [9] Below, we call a control policy that chooses $0 \leq b(t) \leq d(t)$, and $P(t) \in \mathcal{P}_{\text{awake}}^{S(t)}$ when $1_{\text{w}}(t) = 1$ and $P(t) = 0$ otherwise a feasible policy. A feasible policy that ensures (9) is called a stabilizing policy. We then use ϕ^* to denote the optimal value of ϕ over all stabilizing policies, i.e., $\phi^* \triangleq \sup_{\Pi} \phi^{\Pi}$, where ϕ^{Π} denotes the average utility under a stabilizing policy Π. Note that due to the finiteness of the battery size, one may not always achieve the optimal value ϕ^*, which characterizes the optimal system utility as the battery size goes to infinity. Nonetheless, ϕ^* provides an upper bound of the optimal utility.

G. Discussion of the Model

Although our model looks similar to the utility maximization model considered in [10], the problem considered in this paper is more challenging. The main complications are imposed by the constraint (7) and the two-timescale operation mode of the node. Specifically, (7) couples the current power allocation action and the future actions, and the two-timescale operation mode couples the sleep/wake decisions and the power allocation actions during the frames. Though [10] resolves the energy-availability problem with a perturbed max-weight approach, it assumes that each network node is always active and focuses on designing power allocation algorithms. This assumption may not closely capture the power consumption profile of mobile devices. Indeed, according to a recent study [22], a mobile device consumes much more power in the idle mode than in the sleep mode. For example, for a Nexus one phone, the phone consumes 24.9mW when asleep, but it consumes 333.9mW when idle (phone call only consumes 746.8mW). Thus, it is necessary to consider sleep/wake scheduling for energy optimization. Moreover, due to hardware constraints, a node typically remains in the sleep/wake mode for a period that is relatively long compared to the time required for packet transmission. Hence a two-timescale model is required.

Since $\overline{U^{(c)}(t)} = 0$ whenever the node is asleep, the first term $\sum_c \overline{U^{(c)}(t)}$ in the utility function (11) can intuitively be viewed as optimizing the product of the fraction of time when the node is awake and the flow utility achieved during the awake period. Also, note that besides mobile devices, our model also applies to modelling a wireless sensor system powered by the energy harvesting technology. In that case, our results can be adopted for power management in such systems.

Finally, we want to remark that the i.i.d. system dynamics assumption is made only for the ease of algorithm and analysis presentation. Our results can be proven under the more general case where the system dynamics are Markovian, using the variable-size drift analysis developed in [27].

Below, for reader convenience, we summarize the notations in the paper in Table I.

III. Steering the Queues

In this section, we present our first algorithm, the Optimal Sleep/wake scheduling Algorithm (OSA) for the problem.

[8] $E(0) = 0$ means that we start with a zero energy level. We can also pre-store energy in the energy queue and initialize $E(0)$ to any finite positive value up to its capacity. The results in the paper will not be affected.

[9] Throughout the paper, we assume that all limits exist.

Notation	Meaning
$\mathbb{T}_m$	Frame m
$\chi(m)$	Demand state
$h(t)$, $e(t)$	Energy state and the amount harvested
π_χ	Probability of having state χ
$1_{\text{w}}(m)$	Sleep/Wake decision
$b(t)$	Amount of workload fulfilled
$U^{(c)}(r)$	Flow c utility
$D(t)$	Disutility
$R^{(c)}(t)$	Amount of type c traffic admitted
$P(t)$	Power consumption
$Q^{(c)}(t)$, $E(t)$	Data queue and energy queue

TABLE I
TABLE OF NOTATIONS

OSA is designed based on the two-timescale Lyapunov optimization technique developed in [17], combined with weight perturbation [18]. The idea of OSA is to construct a multi-slot Lyapunov scheduling algorithm with *perturbed* weights for determining the control actions. Later in Section V, we will extend the OSA algorithm by incorporating system information into the algorithm.

A. The OSA Algorithm

To start, we first choose a *perturbation* value θ (to be specified later). Then, we define a *perturbed* Lyapunov function as follows:

$$L(t) \triangleq \frac{1}{2}\sum_{c\in\mathcal{C}}\big[Q^{(c)}(t)\big]^2 + \frac{1}{2}\big[E(t)-\theta\big]^2. \tag{12}$$

The intuition behind the use of the θ value is that by keeping the Lyapunov function value small, we "push" the $E(t)$ value towards θ. Thus, by carefully choosing the value of θ, we can ensure that the energy queue always has enough energy when the node is awake.

Now denote $Z(t) = (Q(t), E(t))$ and define a T-slot conditional Lyapunov drift as follows:

$$\Delta_T(t) \triangleq \mathbb{E}\big\{L(t+T) - L(t) \mid Z(t)\big\}. \tag{13}$$

Here the expectation is taken over the randomness of the demand state, the channel state and the energy state, as well as the randomness in choosing the sleep/wake decisions, the data admission action, the power allocation action, the scheduling action, and the energy harvesting action. For notation simplicity, we denote the instantaneous utility as:

$$f(t) \triangleq \sum_c U^{(c)}(1_{\text{w}}(m_t)R^{(c)}(t)) \\ -D(1_{\text{w}}(m_t)b(t), d(t), \chi(m_t)), \tag{14}$$

and the drift-plus-utility as:

$$\Delta_{T,V}(t) \triangleq \Delta_T(t) - \sum_{\tau=t}^{t+T-1}\mathbb{E}\big\{Vf(\tau) \mid Z(t)\big\}. \tag{15}$$

Here in (15), the V parameter is used to control the utility performance of the algorithm. As we will see, V also determines the required capacity of the energy battery under OSA. We first have the following lemma regarding the drift:

Lemma 1: Let $t = mT$, $m \in \{0,1,2,...\}$. Then, under any feasible sleep/wake action, data admission action, power allocation action that satisfies the energy availability constraint (7), scheduling action, and energy harvesting action that can be implemented at time t, we have:

$$\begin{aligned}\Delta_{T,V}(t) &\leq TB + \sum_{\tau=t}^{t+T-1}\mathbb{E}\big\{(E(t)-\theta)e(\tau) \mid Z(t)\big\} \\ &- \sum_{\tau=t}^{t+T-1}\mathbb{E}\Big\{\sum_c\big[VU^{(c)}(1_{\text{w}}(m_t)R^{(c)}(\tau)) \\ &\qquad -Q^{(c)}(t)1_{\text{w}}(m_t)R^{(c)}(\tau)\big] \mid Z(t)\Big\} \\ &- \sum_{\tau=t}^{t+T-1}\mathbb{E}\Big\{\sum_c 1_{\text{w}}(m_t)\mu^{(c)}(\tau)Q^{(c)}(t) \\ &\qquad +(E(t)-\theta)1_{\text{w}}(m_t)P(\tau) \mid Z(t)\Big\} \\ &+ \sum_{\tau=t}^{t+T-1}\mathbb{E}\big\{\big[VD(\tau) - 1_{\text{w}}(m_t)(E(t)-\theta)b(\tau)\big] \mid Z(t)\big\}.\end{aligned} \tag{16}$$

Here $B = \Theta(1)$ is a constant defined in (47), which is independent of the control parameter V. □

Proof: See Appendix A. ■

Lemma 1 is a technical lemma that provides an upper bound on the Lyapunov drift. Intuitively, the drift tells us how the Lyapunov function value changes according to the actions. As we will see below, the right-hand-side (RHS) of the upper bound is the basis of our algorithm design and analysis.

We now present the OSA algorithm. The idea of the algorithm is to minimize RHS of (16) subject to the energy-availability constraint (7). For ease of presenting our algorithm, we define the following function:

$$\begin{aligned}&D_{\text{tot}}(m_t) \\ &\triangleq \sum_{\tau=t}^{t+T-1}\Big[\sum_c\big[VU^{(c)}(R^{(c)}(\tau)) - Q^{(c)}(t)R^{(c)}(\tau)\big]\Big] \\ &\quad + \sum_{\tau=t}^{t+T-1}\Big[\sum_c\mu^{(c)}(\tau)Q^{(c)}(t) + (E(t)-\theta)P(\tau)\Big] \\ &\quad - \sum_{\tau=t}^{t+T-1}\Big[VD(b(\tau), d(\tau), \chi(m_t)) - (E(t)-\theta)b(\tau)\Big].\end{aligned} \tag{17}$$

Note that $D_{\text{tot}}(m_t)$ roughly denotes the controllable components in the RHS of (16) when $1_{\text{w}}(m_t) = 1$ (except for $e(\tau)$). We now have the algorithm as follows:

Optimal Sleep/wake scheduling Algorithm (OSA): Initialize θ (value to be specified). In frame m_t, perform the following:

- Sleep/Wake Decision (Every Frame): Observe $\chi(m_t)$, $Q^{(c)}(t)$ and $E(t)$, solve:

$$\begin{aligned}\max:\quad & \mathbb{E}\big\{D_{\text{tot}}(m_t)\big\} \\ \text{s.t.}\quad & 0 \leq R^{(c)}(\tau) \leq R_{\max},\ \tau \in \mathbb{T}_{m_t} \\ & P(\tau) \in \mathcal{P}_{\text{awake}}^{(S(\tau))}, 0 \leq b(\tau) \leq d(\tau),\ \tau \in \mathbb{T}_{m_t}\end{aligned} \tag{18}$$

  Here the expectation is taken over $d(t)$ and $S(t)$, and the control variables are the rates $R^{(c)}(\tau)$, the power allocation $P(\tau)$, and the computation fulfilment $b(\tau)$ and the rates $\mu^{(c)}(t)$. Denote the optimal solution by D^*_{tot}. Then, if

$$D^*_{\text{tot}} > -\mathbb{E}\big\{\sum_{\tau=t}^{t+T-1}VD(0, d(\tau), \chi(m_t))\big\}, \tag{19}$$

the node enters the awake mode, i.e., $1_{\text{w}}(m_t) = 1$. Otherwise it sets $1_{\text{w}}(m_t) = 0$ and enters the sleep mode. If the node enters the sleep mode, it sets $R^{(c)}(\tau) = P(\tau) = b(\tau) = 0$ for every $\tau \in \mathbb{T}_{m_t}$. On the other hand, if it enters the awake mode, it does the following for traffic admission, power expenditure, and scheduling for every $\tau \in \mathbb{T}_{m_t}$:

– Data Admission: Choose $R^{(c)}(\tau)$ to be the optimal solution of the following optimization problem:

$$\max: \ VU^{(c)}(r) - Q^{(c)}(t)r, \ \ s.t. \ 0 \leq r \leq R_{\max}. \quad (20)$$

– Power expenditure: Choose $0 \leq b(\tau) \leq d(\tau)$ to minimize:

$$W(b(\tau)) \triangleq VD(b(\tau), d(\tau), \chi(m_t)) - (E(t) - \theta)b(\tau).$$

Define $Q^*(t) \triangleq \max_c Q^{(c)}(t)$. Then, choose $P(\tau) \in \mathcal{P}^{(s_i)}_{\text{awake}}$ to maximize:

$$G(P(\tau)) \triangleq \mu(\tau)Q^*(t) + (E(t) - \theta)P(\tau), \quad (21)$$

subject to the energy availability constraint (7).

– Scheduling: Let $c^* \in \{c : Q^{(c)}(t) = Q^*(t)\}$. Transmit commodity c^* packets with rate $\mu(\tau)$, use idle fill if needed.

Note that the data admission action, the power expenditure action, and the scheduling action are indeed the actions that maximizes (18) given $d(\tau)$ and $S(\tau)$ for all $\tau \in \mathbb{T}_{m_t}$.

- Energy Harvesting (Every Slot): In every timeslot $\tau \in \mathbb{T}_{m_t}$, if $E(t) - \theta < 0$, perform energy harvesting and store the harvested energy during that frame, i.e., $e(\tau) = h(\tau)$; else set $e(\tau) = 0$.
- Queue Update (Every Slot): Update $Q^{(c)}(\tau)$ and $E(\tau)$ according to the dynamics (8) and (10), respectively. $\diamond$

Equation (19) can be viewed as comparing the expected "best" performance one can achieve by turning the node ON, i.e., by setting $1_{\text{w}}(m_t) = 1$ and optimizing the actions accordingly, versus the performance one achieves by entering the sleep mode, where all actions are 0. The intuition is that, if by turning on the node, even in the best case, one cannot outperform the sleep mode (in terms of the value of $D_{\text{tot}}(m_t)$), then the node should enter the sleep mode. Note that in the energy harvesting step of OSA, the node will always perform energy harvesting when the energy volume is less than θ, and rejects the harvestable energy otherwise. Hence, $E(t) \leq \theta + Th_{\max}$ for all t. This is an important feature. It allows us to implement OSA with finite energy storage capacity, i.e., use an energy storage size of $\theta + Th_{\max}$ (below we will assume that OSA is implemented with this energy capacity). In practice, OSA will always harvest energy unless the energy storage is full. It can be shown that in this case, the actual energy we store will always be no less than that under OSA. Hence, all the actions of OSA are valid.

B. Implementation of OSA

We note that OSA does not require any knowledge of the energy state process $h(t)$. This is very useful in practice when knowledge of the energy source may be difficult to obtain. However, we note that OSA does require estimation of the channel state statistics and external demand statistics in order to maximize (18). This is different from previous algorithms for energy harvesting network control, e.g., [10]. In practice, this can often be done via historic information. We will also see in the simulation section that OSA can automatically adapt to the environment when there is a distribution change. In Section V, we develop an algorithm to explicitly take such system information into account.

IV. Performance analysis of OSA

We now present the performance results of the OSA algorithm. Below, recall that the parameter β is the largest first derivative of the utility functions defined in (4) and that α is the maximum growth rate of the disutility. The parameter θ is defined to be:

$$\theta \triangleq V(\beta\delta + \frac{\beta|\mathcal{C}|R_{\max}}{P_{\min}} + \frac{\alpha d_{\max}}{P_{\min}}) + \delta T R_{\max} \quad (22)$$
$$+T(P_{\max} + d_{\max}).$$

Note that the value θ can easily be determined. It only requires knowledge of the maximum derivatives of the utility functions and the power-rate curve, and the maximum power expenditure, and requires no statistical knowledge of system dynamics, e.g., the channel and harvestable energy process. As we will show later, (22) also provides us with an easy way to size our energy storage devices for achieving a utility that is within $O(\epsilon)$ of the optimal, i.e., use energy storage devices of size $O(1/\epsilon)$. The sizing rule also demonstrates the relationship between the energy storage capacity and the elasticity of the external demand.

***Theorem** 1:* Under the OSA algorithm with β and θ defined in (4) and (22), we have the following:

(a) The data queues and the energy queue satisfy the following for all time:

$$0 \leq Q^{(c)}(t) \leq \beta V + TR_{\max}, \ \forall \ c, \quad (23)$$
$$0 \leq E(t) \leq \theta + Th_{\max}. \quad (24)$$

Moreover, if at any time t, the node enters the awake state, we must have $E(t) \geq T(d_{\max} + P_{\max})$.

(b) Under OSA, we have:

$$\phi^{\text{OSA}} \geq \phi^* - \frac{B}{V}. \quad (25)$$

Here ϕ^* is the optimal time average utility of our problem, and $B = \Theta(1)$ is defined in Lemma 1. $\square$

Here we present the proof for (23) and (24). The rest of the proof will be given in Appendix B.

Proof: (Part (a)) First we see that $Q^{(c)}(0) = 0$ for all $c \in \mathcal{C}$ satisfies the bounds in (23). Suppose the bound holds for $Q^{(c)}(t)$. We want to show that they also hold for $Q^{(c)}(t+1)$. In the first case, suppose $Q^{(c)}(t) \leq \beta V$. Then, $Q^{(c)}(\tau) \leq V\beta + TR_{\max}$ for all $\tau \in [t, t+T-1]$. This is so because $R_{\max}$ is the maximum arrival rate in any time slot. Now suppose $\beta V < Q^{(c)}(t) \leq \beta V + TR_{\max}$. From the data admission rule of OSA, we see that $R^{(c)}(\tau) = 0$ for all $\tau \in [t, t+T-1]$. Thus, $Q^{(c)}(\tau) \leq Q^{(c)}(t) \leq V\beta + TR_{\max}$.

Similarly, we see that whenever $E(t) > \theta$, OSA will choose $e(\tau) = 0$ for $\tau \in \mathbb{T}_{m_t}$. Hence, $E(t) \leq \theta + Th_{\max}$ for all t. ■

Two remarks on Theorem 1 are in place. (i) By taking $\epsilon = 1/V$, Part (a) implies that the average data queue size is $O(1/\epsilon)$. Combining this with Part (b), we see that OSA achieves an $[O(\epsilon), O(1/\epsilon)]$ utility-backlog tradeoff for our problem. (ii) Part (a) shows that the energy queue size is deterministically upper bounded by a constant of size $O(1/\epsilon)$. Hence, our result provides an explicit characterization of the size of the energy storage device needed for achieving a desired utility performance.[10]

V. Information-Aided energy management

In the previous sections, we have seen that OSA does not require any knowledge of the energy arrival process $h(t)$ and achieves the $[O(\epsilon), O(1/\epsilon)]$ utility-backlog tradeoff. However, in practice, statistical knowledge can often be obtained, e.g., from history or runtime learning. In this case, it is important to efficiently utilize such information. Thus, in this section, we develop the Information-aided OSA algorithm (IOSA) to incorporate the prior statistical information into system control. Different from OSA, IOSA does not try to provide deterministic bounds for the data and energy buffer sizes. Instead, it guarantees that they are bounded by $O(\log(V)^2)$ with high probability.

A. Information-aided OSA

Suppose now we have access to all the statistical information of the system, i.e., π_χ, π_d, π_s, and λ_h. To construct the IOSA algorithm, we consider the following optimization problem.

$$\max: \quad V\sum_{\chi}\pi_\chi\nu_\chi T\bigg[\sum_c U^{(c)}(r^c_\chi) - \sum_d \pi_d D(b^{(d)}_\chi, d, \chi)\bigg] \tag{26}$$

$$\text{s.t.} \quad \sum_\chi \pi_\chi \nu_\chi T\bigg[r^c_\chi - \sum_s \pi_s\mu^c_s\bigg] \leq 0, \forall c \tag{27}$$

$$\sum_\chi \pi_\chi\nu_\chi T\bigg[\sum_d \pi_d b^{(d)}_\chi + \sum_s \pi_s P^{(s)}_\chi\bigg] = T\lambda_h \tag{28}$$

$$\sum_c \mu^c_s \leq \mu(s, P^{(s)}_\chi), \forall s \tag{29}$$

$$P^{(s)}_\chi \in \mathcal{P}^{(s)}_{\text{awake}}, \forall s, \ P^{(s)}_\chi = 0 \text{ if } \chi = 0 \tag{30}$$

$$0 \leq b^{(d)}_\chi \leq d, \forall d, \text{ and } b^{(d)}_\chi = 0 \text{ if } \chi = 0 \tag{31}$$

$$r^c_\chi \in [0, R_{\max}] \text{ if } \chi = 1, \text{ and } r^c_\chi = 0 \text{ if } \chi = 0 \tag{32}$$

$$\nu_\chi \in [0, 1]. \tag{33}$$

This problem can be interpreted as follows. (i) The objective function (26) represents the long term utility the system achieves, where ν_χ denotes the awake probability under state χ, and r^c_χ and $b^{(d)}_\chi$ denote the flow rate and the fulfillment under condition χ and d. (ii) Constraints (27) and (28) then specify that the service rate must be no smaller than the flow arrival rate and that the energy usage rate must be no more than the harvesting rate. (iii) Constraints (29)-(33) are the feasibility conditions. Intuitively, solving (26) gives us a control policy that can be used to control the system, under which ν_χ denotes the probability to stay awake, and $b^{(d)}_\chi$, r^c_χ, and μ^c_s denote the corresponding actions used under different random states. [11] We now also define the dual function of (26):

$$g(\boldsymbol{\gamma}, \xi) \triangleq \sum_\chi \pi_\chi \sup\bigg\{\nu_\chi TV\sum_c U^{(c)}(r^c_\chi) - \nu_\chi TV\sum_d \pi_d D(b^{(d)}_\chi, d, \chi) - \nu_\chi T\sum_c \gamma_c\big[r^c_\chi - \sum_s \pi_s\mu^c_s\big] - \xi T\big(\nu_\chi\big[\sum_d \pi_d a_\chi b^{(d)}_\chi + \sum_s \pi_s P^{(s)}_\chi\big] - \lambda_h\big)\bigg\}. \tag{34}$$

Here the sup is taken over r^c_χ, μ^c_s, $b^{(d)}_\chi$, $P^{(s)}_\chi$, and μ^c_s. Denote $(\boldsymbol{\gamma}^*, \xi^*)$ an optimal solution of the dual problem, i.e.,

$$\min: \quad g(\boldsymbol{\gamma}, \xi), \quad \text{s.t. } \boldsymbol{\gamma} \succeq \mathbf{0}, \ \xi \in \mathbb{R}. \tag{35}$$

It is known from [27] that optimal value of the dual function, i.e., $g(\boldsymbol{\gamma}^*, \xi^*)$ corresponds to the optimal system utility. It is also known from [28] that $(\boldsymbol{\gamma}^*, \xi^*) = \Theta(V)$ or $(\boldsymbol{\gamma}^*, \xi^*) = 0$. Moreover, it has been shown in [28] that, under many Lyapunov-based online algorithms, the queue sizes will be attracted to the optimal Lagrange multiplier of the dual problem. Hence, the idea of IOSA is to first pre-compute $(\boldsymbol{\gamma}^*, \xi^*)$, and then use the values to substitute the true backlog required. With all the above definitions, we now present the information-aided energy management algorithm.

<u>Information-aided OSA (IOSA)</u>: Find $(\boldsymbol{\gamma}^*, \xi^*)$ by solving (35). Then, run OSA with $\theta = 0$ and with the following modifications.

- In all steps of OSA, replace $Q^{(c)}(t)$ by $\tilde{Q}^{(c)}(t) \triangleq Q^{(c)}(t) + \gamma^*_c - \log(V)^2$ for all c, and replace $E(t)$ by $\tilde{E}(t) \triangleq E(t) + \xi^* - \log(V)^2$.
- At any time t, if the resulting $b(t)$ and $P(t)$ is such that $b(t) + P(t) > E(t)$, consume all power until $E(t) = 0$. However, the node does not try to perform any computation or transmission. If $b(t) > 0$, set $b(t) = 0$ and do not perform any computation. If $P(t) > 0$ and the resulting rate for commodity c is $\mu^{(c)}(t)$, drop all $\mu^{(c)}(t)$ packets. That is, still update $Q^{(c)}(t)$ according to (8) but do not count the dropped packets. Also, update $E(t)$ according to:

$$E(t+1) = \big(E(t) - 1_{\text{w}}(m_t)[b(t) + P(t)]\big)^+ + e(t). \quad \diamond \tag{36}$$

A few remarks about the performance are in place. First, note that with $\theta = 0$, IOSA is not trying to avoid energy underflow as in OSA. Instead, it treats the energy storage as a normal queue and allows underflow events to happen. However, it considers such moments as "stopping moments," in that the computing and transmitting actions are not actually performed. Despite the fact that this can lead to packet

[10]The problem of utility maximization under a pre-specified buffer size is more challenging and is an interesting problem for future research.

[11]Technically speaking, due to the possibly non-concave nature of the disutility function and the possible need for time-sharing control actions, solving (26) may not immediately give us an optimal policy. However, it is sufficient for our algorithm design and analysis.

dropping as well as performance loss, doing so has the benefit of being flexible in algorithm design. Moreover, one can show that such halting events happen rarely under IOSA. Second, replacing the backlog values with the augmented values is an important and novel feature of IOSA compared to previous algorithms developed in [10], [11] and [18]. The optimal multiplier $(\boldsymbol{\gamma}^*, \xi^*)$ specifies how prior information can be incorporated into system control. It turns out that with this way of utilizing system information, one can significantly reduce the required buffer sizes for both data and energy.

Note that we have assumed perfect statistical information is given beforehand. When such information is not available, IOSA can be implemented by dedicating some initial slots for learning the distribution. We will see in the simulation section that, even just with a moderate number of slots, IOSA can achieve good performance and small queue size.

B. Performance of IOSA

In this section, we present the performance result of IOSA under the following system structure property:

Definition 1: The system is called *polyhedral* with parameter $\rho > 0$ if for any $(\boldsymbol{\gamma}, \xi)$ with $\boldsymbol{\gamma} \succeq \mathbf{0}$ and $\xi \in \mathbb{R}$, one has:

$$g(\boldsymbol{\gamma}, \xi) \geq g(\boldsymbol{\gamma}^*, \xi^*) + \rho \|(\boldsymbol{\gamma}, \xi) - (\boldsymbol{\gamma}^*, \xi^*)\|, \quad (37)$$

under all V values. $\Diamond$

Note that (37) typically holds when the system control action set is a finite discrete set [28]. The requirement that (37) holds for all V is also not restricted. In fact, it can be shown that once it holds for any $V > 0$, it holds for all V [28].

Theorem 2: Suppose that $(\boldsymbol{\gamma}^*, \xi^*)$ is the unique optimal for $g(\boldsymbol{\gamma}, \xi)$, that the dual function is polyhedral with parameter $\rho = \Theta(1) > 0$, and that the set of possible queue values is countable. Then, under IOSA with a sufficiently large V, we have:

(a) The data queues and the energy queue satisfy the following:

$$\overline{Q^{(c)}(t)} \leq O(\log(V)^2), \ \forall\, c, \quad \overline{E(t)} \leq O(\log(V)^2). \quad (38)$$

Here $\overline{x(t)} \triangleq \lim_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{x(\tau)\}$. Moreover, there exist constants $H, K, p = \Theta(1)$ such that in steady state:

$$\mathbf{Pr}\{|E(t) - \log(V)^2| > H + Km\} \leq pe^{-m}. \quad (39)$$

(b) Under IOSA, we have:

$$\phi^{\text{IOSA}} \geq \phi^* - O(\frac{1}{V}). \quad (40)$$

Here ϕ^* is the optimal time average utility.

c) The average rate of the dropped packets and the average disutility due to the modification steps are both $O(1/V^{\frac{\log(V)}{2K}})$. $\square$

Proof: See Appendix C. ■

It is important to notice the differences between Theorem 2 and Theorem 1. Under the IOSA algorithm, we do not provide deterministic queueing bounds. However, the probabilistic bounds (39) is strong in that the probability for $E(t)$ to deviate from $\log(V)^2$ decreases exponentially in the deviation distance. Thus, for a large enough V values, energy outage rarely happens, and the energy storage capacity does not need to be large (see also Section VI for simulation results). Part c) also guarantees that the average rate for the packet dropping is very small, i.e., $O(1/V^{\frac{\log(V)}{2K}})$, which is negligible even for moderate V values. Different from the OSA case, here we see that IOSA achieves the $[O(\epsilon), O(\log(1/\epsilon)^2)]$ utility-delay tradeoff with energy buffer size $O(\log(1/\epsilon)^2)$. This clearly illustrates the benefit of utilitzing the statistical information of the system dynamics. We also remark here that the analysis of IOSA requires analyzing the combination of queue underflow probability and drift augmentation, and cannot be done by directly following the typical Lyapunov analysis adopted in [10] and [11] due to the incorporation of $(\boldsymbol{\gamma}^*, \xi^*)$.

VI. Simulation

In this section, we simulate the algorithms. To make the setting relevant, we choose our parameters to be normalized versions of those given in [22] for Nexus one. Specifically, in [22] it is shown that the Nexus one phone consumes 333.9mW for idle, 322.4mW for audio, 746mW for phone call, 825mW for cellular network, and 884.1mW for WiFi network. We thus set the idle power to be 1, i.e, $P_{\min} = 1$. Then, we also approximately normalize the audio power, the phone call power and the network power to be 1, 2, and 3. Thus, $\mathcal{P}_{\text{awake}}^{ON} = \mathcal{P}_{\text{awake}}^{OFF} = \{1, 2, 3\}$. Also, $P_{\min} = 1$ and $P_{\max} = 3$.

Each frame consists of $T = 10$ slots. In each frame, $\chi(m) = 1$ with probability 0.6. When $\chi(m_t) = 1$, $d(t)$ takes value uniformly in $\{0, 1, 2, 3\}$ every time. When $\chi(m_t) = 0$, $d(t) = 0$ for all $t \in \mathbb{T}_{m_t}$. The disutility is assumed to take the form (2) with $a = \frac{1}{9}$, i.e.,

$$D(b(t), d(t), \chi(m_t)) = \frac{1}{9}\chi(m_t)(d(t) - b(t))^2, \quad (41)$$

which means $\alpha = 2/3$. The channel state $S(t)$ takes value in $\{0, 1\}$ equally likely. $\mu(t) = \log(1 + 2S(t)(P(t) - P_{\min}))$. Hence we see that $\delta = 2$. We assume that $h(t)$ is also a Bernoulli random variable, which takes value 2 with probability 0.5 and 0 otherwise. We assume there is only one commodity and $U(r) = \log(1 + 2r)$,[12] which means $\beta = 2$. Also, $R(t) \in [0, 2]$ and $R_{\max} = 2$.

With the above parameters, we set $\theta = 10V + 100$. We simulate both OSA and IOSA for $V \in \{10, 20, 40, 80, 100, 150, 200\}$. The results are plotted in Fig. 2. We see that as the V parameter increases, the average utility performance under both schemes increases and quickly converges to the optimal. We also observe that the utility performance of IOSA is very similar to the OSA at all V values, despite the fact that under IOSA energy outage can cause utility loss (the modification steps). Such a utility loss vanishes when V is of moderate size, in which case energy outage rarely happens, i.e., $O(1/V^{\frac{\log(V)}{2K}})$.

We also see from the middle and right plots that the average data queue size and the average energy level increases linear in V under OSA, as per Theorem 1. On the other hand, they grow

[12]Such logarithmic functions are commonly adopted in network utility maximization, e.g., [29].

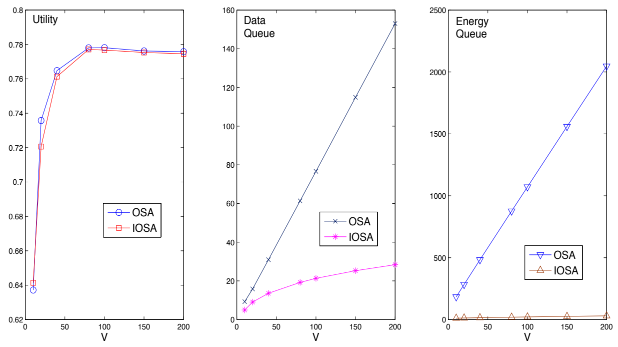

Fig. 2. Average utility, average data queue size and average energy queue siz

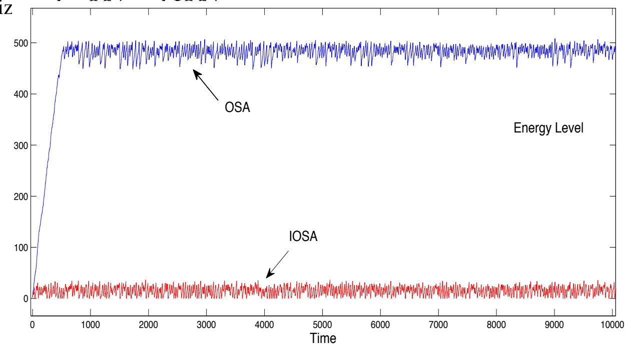

Fig. 3. A sample path energy level process from time 1 to 10000 under $V = 40$.

only poly-logarithmically under IOSA, as shown in Theorem 2. This demonstrates the effect of information.

Fig. 3 shows two sample path energy level processes under OSA and IOSA, respectively. To make the comparison fair, we have run both algorithms under the same sequence of random events and repeated the test multiple times to verify its typicality. Here we also see that IOSA is very effective in reducing the required energy storage capacity. This is enabled by the introduction of $\tilde{\gamma}$ and $\tilde{\xi}$, which substitute the need for building up the true energy level for decision making. We also see from Fig. 3 that the maximum energy level is consistent with our analytical results, i.e., Theorem 1 and 2.

We see that the energy level under OSA never goes below zero and hence all the power expenditure actions are feasible. It can also be verified that the energy level never exceeds the bound in (24). Under IOSA, however, the energy level can become zero. However, one can see that such events happen rarely, as shown in Fig. 4. For instance, when $V = 80$, the outage probability is only 0.002. This is consistent with Part c) of Theorem 2.

Fig. 5 also shows how the sleep/wake decision changes in reaction to the energy level change. We see that OSA is able to adaptively decide its sleep/wake action *without* any statistical knowledge of the harvestable energy process. We omit the results for IOSA as they look very similar.

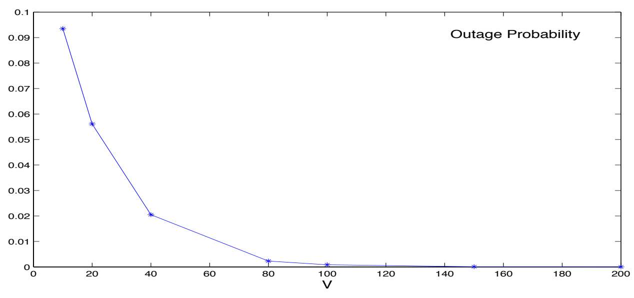

Fig. 4. A sample path energy level process from time 1 to 10000 under $V = 40$.

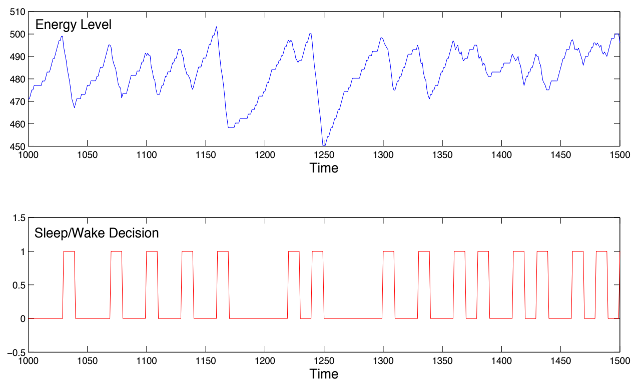

Fig. 5. A sample path energy level process and a sample path sleep/wake decision process under $V = 40$ from time 1000 to 1500.

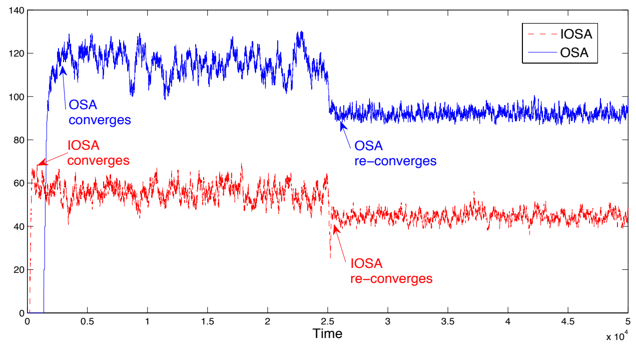

Fig. 6. A sample path energy level process and a sample path sleep/wake decision process under $V = 150$ from 10^5 slots.

Moreover, to see how OSA and IOSA adapts to distribution changes, we also conduct simulations where the distribution is changed in the middle of the simulation. Fig. 6 shows that both OSA and IOSA adapts very well to the change. Here we choose to show the data queue size since the energy queue size change is not very significant, due to the perturbation approach, which pulls the energy queue size around θ (similar to Fig. 3). In the simulation of IOSA, we use the first 200 slots of each distribution to learn the distribution. Under the two distributions, we obtain that under OSA, the average total utilities are 0.6423 for the first half and 1.0263 for the second, whereas under IOSA, we get 0.59 and 0.9767, respectively. It can be seen that the performance are similar. However, IOSA achieves it with a much smaller queue size and faster convergence (under both distributions). Here we also note that the delay reduction is not as significant as in the perfect prior information case. This is due to the error in learning and can be improved by spending more time in learning.

We also look at the algorithm performance for more complex settings. We consider a case where $d(t)$ takes values uniformly in $\{0, 1, ..., 5\}$ every time slot, and that the energy arrival can take any value in $\{\frac{n}{5}, n = 0, ..., 10\}$. Moreover,we change the set of feasible power action to be $\mathcal{P}^{ON}_{\text{awake}} = \mathcal{P}^{OFF}_{\text{awake}} = [1, 3]$. Our goal is to see how the algorithms perform under more general settings. Fig. 7 shows the performance of OSA and IOSA under this setting. It is not hard to see that the

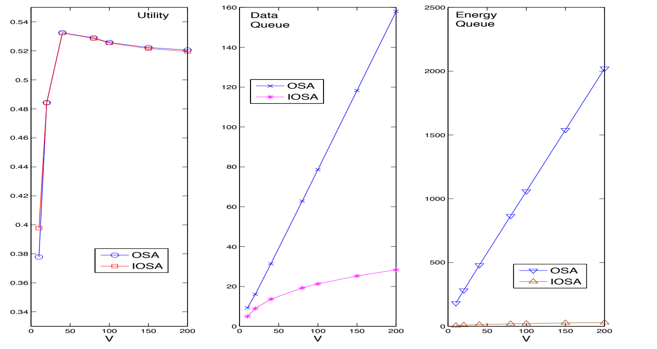

Fig. 7. Average utility, average data queue size and average energy queue size under OSA and IOSA with complex setting.

two algorithms still achieve near optimal performance, while IOSA outperforms OSA in reducing queue sizes significantly. This shows that indeed the two algorithms are quite robust and can also apply to complex setting. The other results of the two algorithms are similar to those in the previous setting, and hence are omitted.

VII. Conclusion

In this paper, we develop two energy management algorithms, i.e., the Optimal Sleep/wake scheduling Algorithm (OSA) and the Information-aided OSA algorithm (IOSA), for achieving optimal system utility for energy harvesting smart mobile devices powered by batteries. OSA is an online algorithm and does not require any knowledge of the harvestable energy processes, whereas IOSA efficiently incorporates system information into control algorithm design. We show that OSA can achieve an $[O(\epsilon), O(1/\epsilon)]$ utility-delay tradeoff with an energy battery of $O(1/\epsilon)$ size. On the other hand, IOSA achieves an $[O(\epsilon), O(\log(1/\epsilon)^2)]$ utility-delay tradeoff and guarantees that using energy storage capacity of $O(\log(1/\epsilon)^2)$ ensures a very small energy outage probability.

Appendix A – Proof of Lemma 1

Here we prove Lemma 1.

Proof: Squaring both sides of (8), summing over $c \in \mathcal{C}$, and multiplying both sides by $\frac{1}{2}$, we obtain:

$$\frac{1}{2}\sum_c \big([Q^{(c)}(\tau+1)]^2 - [Q^{(c)}(\tau)]^2\big) \le \frac{1}{2}\sum_c \big([(R^{(c)}(\tau)]^2 + [\mu^{(c)}(\tau))]^2\big) - 1_{\mathrm{w}}(m_\tau)\sum_c Q^{(c)}(\tau)\big[\mu^{(c)}(\tau) - R^{(c)}(\tau)\big]. \tag{42}$$

Similarly, using (10), we have:

$$\frac{1}{2}\big([E(\tau+1)-\theta]^2 - [E(\tau)-\theta]^2\big) \le \frac{1}{2}[b(\tau)+P(\tau)]^2 + \frac{1}{2}[e(\tau)]^2 - (E(\tau)-\theta)[1_{\mathrm{w}}(\tau_m)(b(\tau)+P(\tau)) - e(\tau)]. \tag{43}$$

Now define:

$$B_1 \triangleq \frac{1}{2}\Big[|\mathcal{C}|(R_{\max}^2 + \mu_{\max}^2) + (d_{\max}+P_{\max})^2 + h_{\max}^2\Big]. \tag{44}$$

Then, by summing (42) over $c \in \mathcal{C}$ and (43), summing over $\tau \in [t, t+T-1]$, taking expectations on both sides conditioning on $Z(t)$, and using the definition of $\Delta_T(t)$, we have:

$$\Delta_T(t) \le B_1 T - \mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}\sum_c Q^{(c)}(\tau)1_{\mathrm{w}}(m_t)[\mu^{(c)}(\tau) - R^{(c)}(\tau)] \mid Z(t)\Big\} - \mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}(E(\tau)-\theta)\big[1_{\mathrm{w}}(m_t)[b(\tau)+P(\tau)] - e(\tau)\big] \mid Z(t)\Big\}. \tag{45}$$

Since all the queues in the system have bounded arrival and service rates, for any $t_1 \le t_2$, we have:

$$\begin{aligned} Q^{(c)}(t_2) &\le Q^{(c)}(t_1) + (t_2-t_1)R_{\max},\\ Q^{(c)}(t_2) &\ge Q^{(c)}(t_1) - (t_2-t_1)\mu_{\max},\\ E(t_2) &\le E(t_1) + (t_2-t_1)h_{\max},\\ E(t_2) &\ge E(t_1) - (t_2-t_1)(d_{\max}+P_{\max}). \end{aligned}$$

Using these inequalities in (45), we obtain that:

$$\Delta_T(t) \le BT - \mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}\sum_c Q^{(c)}(t)1_{\mathrm{w}}(m_t)[\mu^{(c)}(\tau) - R^{(c)}(\tau)] \mid Z(t)\Big\} - \mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}(E(t)-\theta)\big[1_{\mathrm{w}}(m_t)[b(\tau)+P(\tau)] - e(\tau)\big] \mid Z(t)\Big\}. \tag{46}$$

Here

$$B \triangleq B_1 + \frac{(|\mathcal{C}|+1)(T-1)}{2}\Big[\mu_{\max}^2 + R_{\max}^2 + h_{\max}^2 + (d_{\max}+P_{\max})^2\Big]. \tag{47}$$

Adding to both sides the term $-\sum_{\tau=t}^{t+T-1}\mathbb{E}\{Vf(\tau) \mid Z(t)\}$, we obtain:

$$\Delta_T(t) - \sum_{\tau=t}^{t+T-1}\mathbb{E}\big\{Vf(\tau) \mid Z(t)\big\} \le BT - \sum_{\tau=t}^{t+T-1}\mathbb{E}\big\{Vf(\tau) \mid Z(t)\big\} - \mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}\sum_c Q^{(c)}(t)1_{\mathrm{w}}(m_t)[\mu^{(c)}(\tau) - R^{(c)}(\tau)] \mid Z(t)\Big\} - \mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}(E(t)-\theta)\big[1_{\mathrm{w}}(m_t)[b(\tau)+P(\tau)] - e(\tau)\big] \mid Z(t)\Big\}. \tag{48}$$

Now using the definitions of $\Delta_{V,T}(t)$ and $f(t)$, and rearranging the terms, we see that the lemma follows. ■

Appendix B – Proof of Theorem 1

In this section, we present the rest of the proof of Theorem 1. In our proof, we will make use of the following theorem, which states that there exists a stationary and randomized policy (which does not take into account the energy-availability constraint and may not be feasible in practice) that makes sleep/wake decisions, allocates power and achieves optimal utility.

Theorem 3: There exists a stationary and randomized policy Π that has the following structure: During each frame m, Π keeps the node awake with certain probability. Then, the node admits traffic, harvests energy, allocates power and schedules packet transmissions purely according to some random functions of the $\chi(m_t), d(t), h(t), S(t)$ state. Finally, Π achieves the following for all $t = mT, m = 0, 1, ...$

$$\mathbb{E}\{\sum_{\tau=t}^{t+T-1} f^{\Pi}(\tau)\} = T\phi^* \quad (49)$$

$$\mathbb{E}\{1_{\text{w}}^{\Pi}(m_t) \sum_{\tau=t}^{t+T-1} \big[\mu^{(c)\Pi}(\tau) - R^{(c)\Pi}(\tau)\big]\} = 0 \quad (50)$$

$$\mathbb{E}\{\sum_{\tau=t}^{t+T-1} \big[1_{\text{w}}^{\Pi}(m_t)[b^{\Pi}(\tau) + P^{\Pi}(\tau)] - e^{\Pi}(\tau)\big]\} = 0. \square \quad (51)$$

Proof: One can first prove that the there exists a policy that achieves (49), and achieves (50) with "$\geq$" and (51) with "$\leq$," using arguments similar to that in [26]. The theorem can then be proven by noticing that one can always admit a little bit more traffic (as $R_{\max} \geq \mu_{\max}$) and harvest a little bit less energy. ■

Different from previous Lyapunov algorithm analysis, we cannot directly compare the drift value under OSA with that under the above policy. This is because the above policy does not take into account the energy-available constraint when making decisions, while OSA explicitly considers the constraint and hence there may be correlations among actions. Thus, our first step in the proof is to show that the energy-availability constraint is indeed redundant under the OSA algorithm. This step is critical for our analysis and allows us to apply the Lyapunov drift analysis approach [17]. We also note that the analysis here is different from the one in [10]. This is because whenever the node stays awake, it will consume at least $TP_{\min}$ power over a frame. Also, when making the sleep/wake decision, the node actually does not take into account the energy-availability constraint.

Proof: (Part (b)) We first show that whenever $E(t) < T(P_{\max} + d_{\max})$, OSA will put the node into the sleep mode. This claim will then allow us to compare our algorithm with alternative algorithms that chooses control actions *without* taking into account the energy-availability constraint.

To prove the claim, consider a time $t = mT$ and assume that $E(t) < T(P_{\max} + d_{\max})$. Let $R^{(c)*}(\tau)$, $\mu^{(c)*}(\tau)$, $P^*(\tau)$ and $b^*(\tau)$, where $\tau \in \mathbb{T}_{m_t}$, be the optimal solution of (18) for the given $E(t)$ and $\boldsymbol{Q}(t)$. Then, using (4), (5) and (23), we have:

$$\begin{aligned} D_{\text{tot}}(m_t) &\leq T|\mathcal{C}|V\beta R_{\max} \quad (52)\\ &+ \sum_{\tau=t}^{t+T-1} \Big[(V\beta + TR_{\max})\delta P^*(\tau) + (E(t)-\theta)P^*(\tau)\Big] \\ &- \sum_{\tau=t}^{t+T-1} VD(0, d(\tau), \chi(m_t)) + \sum_{\tau=t}^{t+T-1} (E(t)-\theta)b^*(\tau) \\ &+ \sum_{\tau=t}^{t+T-1} V\big[D(0, d(\tau), \chi(m_t)) - D(b(\tau)^*, d(\tau), \chi(m_t))\big]. \end{aligned}$$

Using the definition of θ in (22), we see that

$$E(t) - \theta + (V\beta + TR_{\max})\delta < 0.$$

Hence, $P^*(\tau) = P_{\min}$ for all $\tau \in \mathbb{T}_{m_t}$. Now since the disutility increases no faster than α, i.e., (3), we get that:

$$\sum_{\tau=t}^{t+T-1} V\big[D(0, d(\tau), \chi(m_t)) - D(b(\tau)^*, d(\tau), \chi(m_t))\big] \leq VT\alpha d_{\max}.$$

Using this and the fact that $\sum_{\tau=t}^{t+T-1}(E(t)-\theta)b(\tau) \leq 0$ in (52), we obtain:

$$\begin{aligned} D_{\text{tot}}(m_t) &\leq -\sum_{\tau=t}^{t+T-1} VD(0, d(\tau), \chi(m_t)) \\ &+T|\mathcal{C}|V\beta R_{\max} + VT\alpha d_{\max} + TP_{\min}(V\beta + TR_{\max})\delta \\ &-TP_{\min}\Big[V\beta\delta + \frac{V\beta|\mathcal{C}|R_{\max}}{P_{\min}} + \frac{V\alpha d_{\max}}{P_{\min}} + \delta TR_{\max}\Big] \\ &\leq -\sum_{\tau=t}^{t+T-1} VD(0, d(\tau), \chi(m_t)). \end{aligned}$$

Hence, the node will enter the sleep mode according to OSA. This shows that whenever the node stays awake, it has enough energy for the whole frame. Thus, the energy-availability constraint is indeed redundant in the OSA algorithm. Hence, though OSA explicit considers the constraint (7) in the power expenditure step, it remains the same even if the constraint is removed.

Having established this property, we see from the control rules that OSA minimizes the RHS of the drift inequality (16) over all policies, including those that do not consider the energy-availability constraint. Hence, the inequality remains valid if we plug in any alternative control policy that makes two-stage decisions. In particular, we plug in the policy Π in Theorem 3 into (48) to get:

$$\Delta_T(t) - \sum_{\tau=t}^{t+T-1} \mathbb{E}\{Vf^{\text{OSA}}(\tau) \mid Z(t)\} \leq BT - VT\phi^*. \quad (53)$$

Taking expectations over $Z(t)$ on both sides, and taking a telescoping sum over $t = mT, m = 0, ..., M-1$, we have:

$$\mathbb{E}\{L(MT) - L(0)\} - \sum_{\tau=0}^{MT-1} \mathbb{E}\{Vf^{\text{OSA}}(\tau)\} \quad (54)$$
$$\leq BMT - VMT\phi^*.$$

Dividing both sides by MTV, taking a limit as $M \to \infty$, and using the fact that $\mathbb{E}\{L(0)\} < \infty$, we have:

$$\lim_{M\to\infty} \frac{1}{MT} \sum_{\tau=0}^{MT-1} \mathbb{E}\{f^{\text{OSA}}(\tau)\} \geq \phi^* - \frac{B}{V}.$$

Using the definition of $f(\tau)$, we get:

$$\lim_{M\to\infty} \frac{1}{MT} \sum_{\tau=0}^{MT-1} \mathbb{E}\{\sum_c U^{(c)}(R^{(c)}(\tau)) - D(\tau)\} \geq \phi^* - \frac{B}{V}.$$

Thus, we conclude that:

$$\phi^{\text{OSA}} = \sum_c \overline{U^{(c)}(R^{(c)}(\tau))} - \overline{D} \geq \phi^* - \frac{B}{V}.$$

This completes the proof of Part (b). ■

Appendix C – Proof of Theorem 2

Here we prove Theorem 2. To do so, we need the following theorem from [28].

***Theorem** 4:* Suppose that $(\gamma^*, \xi^*) > 0$ is the unique optimal for $g(\boldsymbol{\gamma}, \xi)$, that the dual function is polyhedral with parameter $\rho = \Theta(1) > 0$. Then, under OSA with $\theta = 0$ (assuming that the energy queue can go negative), there exist constants $H = \Theta(1)$ and $\eta = \Theta(1) > 0$, such that whenever $||(\boldsymbol{Q}(t), E(t)) - (\boldsymbol{\gamma}^*, \xi^*)|| > H$,

$$\mathbb{E}\big\{||(\boldsymbol{Q}(t+1), E(t+1)) - (\boldsymbol{\gamma}^*, \xi^*)|| \mid \boldsymbol{Q}(t), E(t)\big\} \qquad (55)$$
$$\leq ||(\boldsymbol{Q}(t), E(t)) - (\boldsymbol{\gamma}^*, \xi^*)|| - \eta.$$

Proof: See [28]. ■

Now we present the proof of Theorem 2.

Proof: (Theorem 2) (Part a)) Note that IOSA can be viewed as OSA with $\theta = 0$ and that the effective queue sizes being $\tilde{\boldsymbol{Q}}(t)$ and $\tilde{E}(t)$. Therefore, using (55), we see that whenever $||(\tilde{\boldsymbol{Q}}(t), \tilde{E}(t)) - (\boldsymbol{\gamma}^*, \xi^*)|| > H$, we have:

$$\mathbb{E}\big\{||(\tilde{\boldsymbol{Q}}(t+1), \tilde{E}(t+1)) - (\boldsymbol{\gamma}^*, \xi^*)|| \mid \tilde{\boldsymbol{Q}}(t), \tilde{E}(t)\big\} \qquad (56)$$
$$\leq ||(\tilde{\boldsymbol{Q}}(t), \tilde{E}(t)) - (\boldsymbol{\gamma}^*, \xi^*)|| - \eta.$$

Or equivalently,

$$\mathbb{E}\big\{||(\boldsymbol{Q}(t+1), E(t+1)) - \log(V)^2 \cdot \mathbf{1}|| \mid \tilde{\boldsymbol{Q}}(t), \tilde{E}(t)\big\} \qquad (57)$$
$$\leq ||(\boldsymbol{Q}(t), E(t)) - \log(V)^2 \cdot \mathbf{1}|| - \eta.$$

Having established (57), one can then apply a similar argument as in the proof of Theorem 1 in [28] to show that, for a sufficiently large V satisfying $\log(V)^2 \geq H$, there exist constants $H, K, p = \Theta(1)$, such that in steady state,

$$\mathbf{Pr}\big\{|\tilde{Q}^{(c)}(t) - \log(V)^2| > H + Km\big\} \leq pe^{-m}. \qquad (58)$$

This implies that the average queue size of $Q^{(c)}(t)$ is $\log(V)^2 + H + O(1) = O(\log(V)^2)$. A similar argument shows that:

$$\mathbf{Pr}\big\{|\tilde{E}(t) - \log(V)^2| > H + Km\big\} \leq pe^{-m}, \qquad (59)$$

which implies that the time average value of $E(t)$ is also $O(\log(V)^2)$.

(Part b)) We now consider the utility performance. We prove our result as follows. First, we prove the overall performance assuming that *there is no energy outage* and ignoring the effect of the modification steps. Then, we come back to refine the result by taking into consideration the energy outage event using (39).

We first consider the case when there is no energy outage. In this case, all actions chosen by IOSA will be executed and the corresponding computation will be fulfilled and the corresponding packets will be sent. To analyze this, we add to both sides of (16) the following term (recall that $\theta = 0$ in IOSA):

$$\varphi_T(t) \qquad (60)$$
$$\triangleq \mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}\sum_c \tilde{\gamma}_c 1_{\text{w}}(m_t)[\mu^{(c)}(\tau) - R^{(c)}(\tau)] \mid Z(t)\Big\}$$
$$+\mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}\tilde{\xi}\big[1_{\text{w}}(m_t)[b(\tau) + P(\tau)] - e(\tau)\big] \mid Z(t)\Big\}.$$

We get:

$$\Delta_{T,V}(t) + \varphi_T(t) \leq TB \qquad (61)$$
$$+\sum_{\tau=t}^{t+T-1}\mathbb{E}\big\{(E(t) + \tilde{\xi})e(\tau) \mid Z(t)\big\}$$
$$-\sum_{\tau=t}^{t+T-1}\mathbb{E}\Big\{\sum_c\big[VU^{(c)}(1_{\text{w}}(m_t)R^{(c)}(\tau))$$
$$-[Q^{(c)}(t) + \tilde{\gamma}_c]1_{\text{w}}(m_t)R^{(c)}(\tau)\big] \mid Z(t)\Big\}$$
$$-\sum_{\tau=t}^{t+T-1}\mathbb{E}\Big\{\sum_c 1_{\text{w}}(m_t)\mu^{(c)}(\tau)[Q^{(c)}(t) + \tilde{\gamma}_c]$$
$$+(E(t) + \tilde{\xi})1_{\text{w}}(m_t)P(\tau) \mid Z(t)\Big\}$$
$$+\sum_{\tau=t}^{t+T-1}\mathbb{E}\big\{\big[VD(\tau) - 1_{\text{w}}(m_t)(E(t) + \tilde{\xi})b(\tau)\big] \mid Z(t)\big\}.$$

Plugging in the optimal randomized policy in Π in Theorem 3, we have:

$$\Delta_{T,V}(t) + \varphi_T(t) \leq TB - VT\phi^*. \qquad (62)$$

Carrying out a telescoping sum over $t = mT$, $m = 0, 1, ..., M-1$, we have:

$$\mathbb{E}\big\{L(MT) - L(0)\big\} - \sum_{\tau=0}^{MT-1}\mathbb{E}\big\{Vf(\tau)\big\} + \sum_{m=0}^{M-1}\varphi_T(mT)$$
$$\leq BMT - VMT\phi^*.$$

Rearranging the terms, dividing both sides by MTV, and taking a $\limsup$ as $M \to \infty$, we have:

$$\limsup_{M\to\infty}\frac{1}{MT}\sum_{\tau=0}^{MT-1}\mathbb{E}\big\{f(\tau)\big\} \qquad (63)$$
$$\geq \phi^* - \frac{B}{V} - \frac{1}{MTV}\sum_{m=0}^{M-1}\varphi_T(mT).$$

It remains to show that the last term $\frac{1}{MTV}\sum_{m=0}^{M-1}\varphi_T(mT) = O(1/V)$. Since both $\tilde{\gamma}_c$ and $\tilde{\xi}$ are $\Theta(V)$, it suffices to show that under IOSA, both the data queues and the energy queue are rarely empty, based on which we can conclude that the difference between the average output and input rates are very small. Hence, from the definition of $\varphi_T(t)$, one can see that the time average value of $\varphi_T(t)$ is small.

To prove this, note that (58) implies that when V is large enough such that $\log(V)^2 - H - K\log(V) \geq \mu_{\max}$, for any commodity c, we have in steady state that:

$$\mathbf{Pr}\big\{Q^{(c)}(t) < \log(V)^2 - H - K\log(V)\big\} \leq pe^{-\log(V)} = \frac{p}{V}.$$

Since at every time slot, the queue can serve at most $\mu_{\max}$ packets, we conclude that:

$$\lim_{T\to\infty}\frac{1}{T}\sum_{\tau=0}^{T-1}\mathbb{E}\big\{1_{\text{w}}(m_t)[\mu^{(c)}(\tau) - R^{(c)}(\tau)]\big\} \leq \frac{\mu_{\max}p}{V}. \qquad (64)$$

A similar argument shows that when $\log(V)^2 - H - K\log(V) \geq 0$, we have:

$$\lim_{T\to\infty}\frac{1}{T}\sum_{\tau=0}^{T-1}\mathbb{E}\big\{1_{\text{w}}(m_t)[b(\tau) + P(\tau)] - e(\tau)]\big\} \qquad (65)$$
$$\leq \frac{(d_{\max} + P_{\max})p}{V}.$$

Using (64) and (65) in (63), we conclude that:

$$\limsup_{M\to\infty}\frac{1}{MT}\sum_{\tau=0}^{MT-1}\mathbb{E}\big\{f(\tau)\big\} \geq \phi^* - O(\frac{1}{V}).$$

This proves the utility performance, assuming no energy outage events. In Part c) below, we will show that the average rate of outage events is $O(1/V)$. Hence, the utility loss is $O(\frac{|\beta\mathcal{C}|+\alpha d_{\max}}{V})$. This completes the proof of Part b).

(Part c)) We now look at the packet dropping part. It can be seen from the IOSA algorithm that packets are dropped only when there is not enough energy in the energy queue. However, when V is large enough to satisfy $\log(V)^2 - H - K[\frac{1}{2K}\log(V)^2] \geq d_{\max} + P_{\max}$, we have in steady state that, the energy outage probability is bounded by:

$$\mathbf{Pr}\big\{E(t) < \log(V)^2 - H - K[\frac{1}{2K}\log(V)^2]\big\} \quad \leq \quad \frac{p}{V^{\frac{\log(V)}{2K}}}.$$

Since in every time slot, there can be at most $\mu_{\max}$ packets dropped from the queue. We conclude that the average packet dropping rate is $O(1/V^{\frac{\log(V)}{2K}})$. Similarly, disutility due to energy outage occurs with probability of no more than $\frac{p}{V^{\frac{\log(V)}{2K}}}$, which results in an average disutility of $O(\frac{\alpha d_{\max} p}{V^{\frac{\log(V)}{2K}}})$. ■

References

[1] S. Perez. The number of mobile devices will exceed worlds population by 2012 (and other shocking figures). *TechCrunch Article, available at http://techcrunch.com/2012/02/14/the-number-of-mobile-devices-will-exceed-worlds-population-by-2012-other-shocking-figures*, February 14, 2012.

[2] D. Etherington. Android phones and tablets ranked by battery life: Longest lasting smartphones arent top-tier devices. *TechCrunch Article, available at http://techcrunch.com/2012/12/18/android-phones-and-tablets-ranked-by-battery-life-longest-lasting-smartphones-arent-top-tier-devices*, December 18, 2012.

[3] M. Gorlatova, P. Kinget, I. Kymissis, D. Rubenstein, X. Wang, and G. Zussman. Challenge: Ultra-low-power energy-harvesting active networked tags (EnHANTs). *Proceedings of ACM MobiCom*, Sept. 2009.

[4] S. Meninger, J. O. Mur-Miranda, R. Amirtharajah, A. Chandrakasan, and J. H. Lang. Vibration-to-eletric energy conversion. *IEEE Trans. on VLSI, Vol. 9, No.1*, Feb. 2001.

[5] V. Raghunathan, A. Kansal, J. Hsu, J. Friedman, and M. B. Srivastava. Design considerations for solar energy harvesting wireless embedded systems. *Proceedings of IEEE IPSN*, April 2005.

[6] V. Sharma, U. Mukherji, V. Joseph, and S. Gupta. Optimal energy management policies for energy harvesting sensor nodes. *IEEE Trans. on Wireless Communication, Vol.9, Issue 4.*, April 2010.

[7] R. Srivastava and C. E. Koksal. Basic tradeoffs for energy management in rechargeable sensor networks. *ArXiv Techreport arXiv: 1009.0569v1*, Sept. 2010.

[8] M. Gatzianas, L. Georgiadis, and L. Tassiulas. Control of wireless networks with rechargeable batteries. *IEEE Trans. on Wireless Communications, Vol. 9, No. 2*, Feb. 2010.

[9] L. Lin, N. B. Shroff, and R. Srikant. Asymptotically optimal power-aware routing for multihop wireless networks with renewable energy sources. *Proceedings of INFOCOM*, 2005.

[10] L. Huang and M. J. Neely. Utility optimal scheduling in energy harvesting networks. *Proceedings of ACM MobiHoc*, May 2011.

[11] C. Tapparello, O. Simeone, and M. Rossi. Dynamic compression-transmission for energy-harvesting multihop networks with correlated sources. *IEEE/ACM Trans. on Networking, Vol. 22, No. 6*, Dec 2014.

[12] V. Joseph, V. Sharma, and U. Mukherji. Optimal sleep-wake policies for an energy harvesting sensor node. *Proceedings of IEEE International Conference on Communications*, June 2009.

[13] N. Michelusi, K. Stamatiou, and M. Zorzi. Transmission policies for energy harvesting sensors with time-correlated energy supply. *IEEE Transactions on Communications, Vol. 61, No. 7*, 2013.

[14] K. J. Prabuchandran, S. K. Meena, and S. Bhatnagar. Q-learning based energy management policies for a single sensor node with finite buffer. *IEEE Wireless Comm. Letters*, February 2013.

[15] C. Hsu, C. Liu, and H. Wang. A reinforcement learning-based tod provisioning dynamic power management for sustainable operation of energy harvesting wireless sensor node. *IEEE Trans. on Emerging Topics in Computing*, April 2014.

[16] S. Fahmy Y. Wu and N. B. Shroff. Optimal sleep/wake scheduling for time-synchronized sensor networks with qos guarantees. *IEEE/ACM Trans. on Networking, vol. 17, Issue 5, pp. 1508-1521.*, October 2009.

[17] 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. 1-144, 2006.

[18] L. Huang and M. J. Neely. Utility optimal scheduling in processing networks. *Proceedings of IFIP Performance*, 2011.

[19] M. J. Neely and L. Huang. Dynamic product assembly and inventory control for maximum profit. *Proceedings of IEEE Conference on Decision and Control*, Dec. 2010.

[20] C. M. Vigorito, D. Ganesan, and A.G. Barto. Adaptive duty cycling for energy harvesting systems. *Proceedings of ACM ISLPED*, 2006.

[21] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava. Power management in energy harvesting sensor networks. *ACM Trans. on Embedded Computing Systems, Vol.6, Issue 4*, Sept. 2007.

[22] A. Carroll and G. Heiser. An analysis of power consumption in a smartphone. *Proceedings of the USENIX conference*, 2010.

[23] N. Li, L. Chen, and S. H. Low. Optimal demand response based on utility maximization in power networks. *IEEE Power and Energy Society General Meeting*, 2011.

[24] G. Girish, D. Riess, and M. Ann Piette. Analysis of open automated demand response deployments in california and guidelines to transition to industry standards. *LAWRENCE BERKELEY NATIONAL LABORATORY Reports LBNL-6560E*, Jan 2014.

[25] L. Huang, J. Walrand, and K. Ramchandran. Optimal demand response with energy storage management. *Proceedings of IEEE SmartGridComm*, November 2012.

[26] M. J. Neely. Energy optimal control for time-varying wireless networks. *IEEE Transactions on Information Theory 52(7): 2915-2934*, July 2006.

[27] L. Huang and M. J. Neely. Max-weight achieves the exact $[O(1/V), O(V)]$ utility-delay tradeoff under Markov dynamics. *arXiv:1008.0200v1*, 2010.

[28] L. Huang and M. J. Neely. Delay reduction via Lagrange multipliers in stochastic network optimization. *IEEE Trans. on Automatic Control*, 56(4):842–857, April 2011.

[29] F. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, vol. 8, pp. 33-37, 1997.