Optimal Sleep-Wake Scheduling for Energy Harvesting Smart Mobile Devices

Longbo Huang
Institute for Interdisciplinary Information Sciences, Tsinghua University
longbohuang@tsinghua.edu.cn

***Abstract*—In this paper, we develop optimal sleep/wake scheduling algorithms for smart mobile devices that are powered by finite capacity batteries and are capable of harvesting energy from the environment. Using a novel combination of the two-timescale Lyapunov optimization approach and the weight perturbation technique, we design the Optimal Sleep/wake scheduling Algorithm (OSA), which *does not* require any statistical 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 $O(1/\epsilon)$.**

I. Introduction

According to a recent report [1], the number of smart mobile devices will soon exceed the world's population. With the rapidly increasing computing power, these mobile smart devices will become a very important component of our daily computing resource. However, 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 newly developed energy harvesting technologies can likely be a remedy for the poor battery performance of the 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, each consists 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 computing demand, the system also supports data transmission (or delay-tolerant workload processing). Every frame, the first decision the device has to make is to decide whether to enter the sleep mode or to stay awake during the frame. If it stays awake, then in every time slot of the frame, it determines how much computing task 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 task demand. The objective of the system 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 met 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. Such problems 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 system size 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] considers the problem of optimal power management for sensor nodes, under the assumption that the harvested energy satisfies a leaky-bucket type property. [8] looks at the problem of designing energy-efficient schemes for maximizing the decay exponent of the queue length. [9] develops scheduling algorithms to achieve close-to-optimal utility for energy harvesting networks with time varying channels. [10] develops an energy-aware routing scheme that approaches optimal as the network size increases. [11] designs optimal control schemes for general multihop energy harvesting networks. [12] constructs optimal sleep/wake schemes for a single node system. [13] considers balancing energy and latency in a sensor network under Markovian system models. However, most of the aforementioned works assume that the nodes in the system always remain "on" and only make power allocation decisions, whereas in practice, nodes typically go through sleep/wake cycles [14]. Hence, the previous results do not consider this two-timescale operation pattern of the network nodes and are not directly applicable.

We tackle this problem using a novel combination of the two-timescale Lyapunov optimization technique developed in [15] and the idea of *weight perturbation* developed in [16] and [17]. The idea of this approach is to construct the algorithm based on a multi-slot quadratic Lyapunov function, but carefully perturb the weights used for decision making, so

as to "push" the target energy levels towards certain nonzero values to avoid energy outage. Based on this approach, we construct the Optimal Sleep/wake scheduling Algorithm (OSA) for achieving optimal utility for smart mobile devices with energy harvesting capabilities and are powered by finite capacity energy batteries. OSA is an *online* algorithm which makes greedy decisions every frame and *does not require any statistical information 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 energy storage devices that are of $O(1/\epsilon)$ sizes. 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 paper is mostly related to the recent works [11] and [9], 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. Hence, the problem considered in this paper can be viewed as a generalization of the problems studied in [9] and [11]. Also, since our algorithm is constructed based on a two-timescale approach, it is very different from the previous algorithms in [9] and [11].

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 results of the OSA algorithm are presented in Section IV. Simulation results are presented in Section V. We conclude the paper in Section VI.

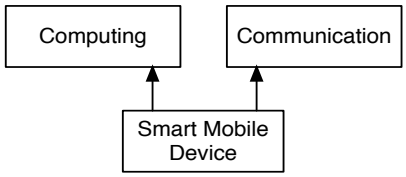

Fig. 1. The system model. The smart mobile device provides computing and communication services to its users or software applications.

II. The System Model

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

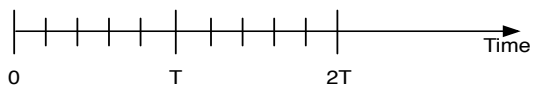

Fig. 2. The time slot and frame structure.

A. The Demand State and the 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 $1_{\rm w}(m_t)$, where $m_t = \lfloor t/T \rfloor$. That is, $1_{\rm w}(m_t) = 1$ if the node stays awake during the frame slot t belongs to, otherwise $1_{\rm w}(m_t) = 0$. However, in some frames, the node may receive external requests to perform certain tasks, e.g., a smartphone user wants to perform some computing task or a software application requires some processing power. To model such situations, 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 $\chi(m) = 1$, then in every time slot $t \in \mathbb{T}_m$, the node receives an external power demand $0 \leq d(t) \leq d_{\max}$. [1] The node then decides how much power to allocate to fulfill the request. 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, then $b(t) = 0$ for all $t \in \mathbb{T}_{m_t}$. [2] Finally, we use

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

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

$$D(t) = a\chi(m_t)(1_{\rm w}(m_t)b(t) - d(t))^2. \quad (2)$$

Here $a > 0$ 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 disutility due to partial fulfillment. We also assume that for any $0 \leq b_1, b_2 \leq d$, there exists a constant $\alpha > 0$ such that:

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

That is, the disutility growth rate is no larger than α, which indicates the degree of elasticity of the user or the software requesting power. A larger α implies that the user is less willing to accept partial fulfillment. Note that such partial fulfillment can be viewed as the node is performing demand response [18]. Similar scenarios already exist in today's smartphones, where the phone reminds the user about the energy level when receiving computing requests.

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

B. The Traffic Utility Model

Besides satisfying external power demands, the node also provides data delivery service to a set of flows (called commodities) denoted by $\mathcal{C}$, e.g., file transfer for different software

[1] Without loss of generality, here we model all the external tasks purely by the power they consume.

[2] 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 by 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 according to the battery level.

applications. [3] 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 $U^{(c)}(\overline{r}^c)$, where $\overline{r}^c$ is the time average rate of the commodity c traffic admitted into node n, defined as $\overline{r}^c = \lim_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{R^{(c)}(\tau)\}$. [4] 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 i.i.d. every slot. At every time slot, if $S(t) = s_i$ and the node stays awake, then 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 $P_{\min} \leq P(t) \leq P_{\max}$ for some $P_{\min} > 0$ 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. 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:

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

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 [19].

We also assume 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 $S(t)$. 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 *finite* capacity energy battery. We model the battery using an *energy queue*, denoted by $E(t)$, which measures the amount of the energy left in the battery at time t. We assume the node can observe its remaining energy level $E(t)$. In any time slot t, the power consumption 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.

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, 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 $\mathcal{H} = \{h_1, ..., h_{M_h}\}$. We assume that the pair $[h(t), S(t)]$ is i.i.d. over slots (possibly correlated in the same slot), with distribution $\pi(h_i, s_j)$ and marginals $\pi(h_i)$, $\pi(s_j)$, respectively.

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 that each energy queue has infinite capacity, and that each node can decide whether or not to harvest energy in each slot. [6] 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. We will show later that our algorithm results in a deterministic energy storage bound, hence can easily be implemented with finite capacity batteries. Note here we assume that the energy harvesting action is not affected by the sleep/wake mode of the node.

E. Queueing Dynamics

Let $Q(t) = (Q^{(c)}(t), c \in \mathcal{C})$, $t = 0, 1, 2, ...$ be the data queue backlog vector in the node, where $Q^{(c)}(t)$ is the amount of commodity c data. We assume the following queueing dynamics:

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

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 the following holds:

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

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:

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

[3] Here we only consider the case when the other major responsibility of the node is serving traffic flows. These flows can also be used to model computing workload that are delay tolerant.

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

[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 will discuss the implementation of our algorithm in Section III-A.

with $E(0) = 0$. [7] By using the queueing dynamic (10), we start by assuming that each energy queue has infinite capacity. Later, we will show that under our algorithm, the energy level $E(t)$ is *deterministically* upper bounded, thus we only need a finite energy capacity for algorithm implementation.

F. Utility Maximization with Energy Management

The goal of the network is thus to design a joint sleep/wake management, flow control, routing and 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 $R^{(c)}(t)$, fulfills 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.,

$$\phi(\overline{\boldsymbol{r}}, \overline{D}) = \sum_{c} U^{(c)}(\overline{r}^c) - \overline{D}, \tag{11}$$

subject to the system stability constraint (9). Here $\overline{\boldsymbol{r}} = (\overline{r}^c, \forall\, c \in \mathcal{C})$ is the vector of the average expected admitted rates, and $\overline{D}$ is the time average disutility incurred due to partial fulfillment of the external power demand, i.e.,

$$\overline{D} \triangleq \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\big\{D(\tau)\big\}. \tag{12}$$

Below, we call a control policy that chooses $0 \leq b(t) \leq d(t)$, and $P(t) \in \mathcal{P}^{S(t)}_{\text{awake}}$ 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 $\phi(\overline{\boldsymbol{r}}, \overline{D})$ over all stabilizing policies and let $\boldsymbol{r}^* = (r^{c*}, \forall\, c \in \mathcal{C})$ to denote the optimal rate vector.

G. Discussion of the Model

Although our model looks similar to the utility maximization model considered in [11], [15], and [20], the problem considered in this paper is more complicated. The main complication is imposed by the constraint (7) and the two-timescale operation mode of the node. Specifically, the constraint (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 [11] resolves the energy-availabilty problem with a perturbed max-weight approach, it assumes that each network node is always active and focuses on designing power allocation algorithms.

III. Engineering the Queues

In this section, we present the Optimal Sleep/wake scheduling Algorithm (OSA) for our problem. OSA is designed based on the two-timescale Lyapunov optimization technique developed in [15], combined with weight perturbation [16]. Thus, it can be viewed as extending the ESA algorithm developed in [11]. The idea of OSA is to construct a multi-slot Lyapunov scheduling algorithm with *perturbed* weights for determining the sleep/wake control, energy harvesting, power allocation, routing and scheduling decisions.

[7] $E(0) = 0$ 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.

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:

$$L(t) \triangleq \frac{1}{2}\sum_{c\in\mathcal{C}} \big[Q^{(c)}(t)\big]^2 + \frac{1}{2}\big[E(t) - \theta\big]^2. \tag{13}$$

The intuition behind the use of the θ value is that by keeping the Lyapunov function value small, we indeed "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:

$$\Delta_T(t) \triangleq \mathbb{E}\big\{L(t+T) - L(t) \mid Z(t)\big\}. \tag{14}$$

Here the expectation is taken over the randomness of the demand state, the channel state and the energy state, as well as the possible 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 define the instantaneous utility:

$$f(t) \triangleq \sum_{c} U^{(c)}(1_{\text{w}}(m_t)R^{(c)}(t)) - D(1_{\text{w}}(m_t)b(t), d(t), \chi(m_t)), \tag{15}$$

and the drift-plus-utility as:

$$\Delta_{T,V}(t) \triangleq \Delta_T(t) - \sum_{\tau=t}^{t+T-1} \mathbb{E}\big\{Vf(\tau) \mid Z(t)\big\}. \tag{16}$$

Here in (16), 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.

We now first have the following lemma regarding the drift:

Lemma 1: Let $t = mT$, $m \in \{0, 1, 2, ...\}$. 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 in $[t, t+T-1]$, we have:

$$\begin{aligned}\Delta_{T,V}(t) \leq TB &+ \sum_{\tau=t}^{t+T-1} \mathbb{E}\big\{(E(t)-\theta)e(\tau) \mid Z(t)\big\} \\ &- \sum_{\tau=t}^{t+T-1} \mathbb{E}\Big\{\sum_{c}\big[VU^{(c)}(1_{\text{w}}(m_t)R^{(c)}(\tau)) \\ &\qquad - Q^{(c)}(t)1_{\text{w}}(m_t)R^{(c)}(\tau)\big] \mid Z(t)\Big\} \\ &- \sum_{\tau=t}^{t+T-1} \mathbb{E}\Big\{\sum_{c} 1_{\text{w}}(m_t)\mu^{(c)}(\tau)Q^{(c)}(t) \\ &\qquad + (E(t)-\theta)1_{\text{w}}(m_t)P(\tau) \mid Z(t)\Big\} \\ &+ \sum_{\tau=t}^{t+T-1} \mathbb{E}\big\{\big[VD(\tau) - 1_{\text{w}}(m_t)(E(t)-\theta)b(\tau)\big] \mid Z(t)\big\}.\end{aligned} \tag{17}$$

Here $B = \Theta(T)$ is a constant defined in (33), which is independent of the control parameter V. □

Proof: See Appendix A. ■

We now present the OSA algorithm. The idea of the algorithm is to minimize the right-hand-side (RHS) of (17) subject to the energy-availability constraint (7). In our algorithm

presentation, we use the following metric for determining the sleep/wake decision in each frame:

$$D_{\text{tot}}(m_t) \triangleq \sum_{\tau=t}^{t+T-1}\Big[\sum_c \big[VU^{(c)}(R^{(c)}(\tau)) - Q^{(c)}(t)R^{(c)}(\tau)\Big] + \sum_{\tau=t}^{t+T-1}\Big[\sum_c \mu^{(c)}(\tau)Q^{(c)}(t) + (E(t)-\theta)P(\tau)\Big] - \sum_{\tau=t}^{t+T-1}\Big[VD(b(\tau),d(\tau),\chi(m_t)) - (E(t)-\theta)b(\tau)\Big]. \quad (18)$$

Optimal Sleep/wake scheduling Algorithm (OSA): Initialize θ. In frame m, perform the following:

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

$$\max: \quad \mathbb{E}\big\{D_{\text{tot}}(m_t)\big\} \quad (19)$$
$$\text{s.t.} \quad 0 \le R^{(c)}(\tau) \le R_{\max},\ \tau \in \mathbb{T}_{m_t},$$
$$P(\tau) \in \mathcal{P}^{(S(\tau))}_{\text{awake}}, 0 \le b(\tau) \le d(\tau),\ \tau \in \mathbb{T}_{m_t},$$

  Here the expectation is taken over $d(t)$ and $S(t)$. Denote the optimal solution by D^*_{tot}. Then, if

$$D^*_{\text{tot}} > -\mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1} VD(0,d(\tau),\chi(m_t))\Big\}, \quad (20)$$

  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 all $\tau \in \mathbb{T}_{m_t}$. Else 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:

$$\max:\ VU^{(c)}(r) - Q^{(c)}(t)r,\ \ s.t.\ 0 \le r \le R_{\max}. \quad (21)$$

  - Power expenditure: Choose $0 \le b(\tau) \le d(\tau)$ 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:

$$G(P(\tau)) \triangleq \mu(\tau)Q^*(t) + (E(t)-\theta)P(\tau), \quad (22)$$

    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 the actions that maximize (19) given $d(\tau)$ and $S(\tau)$ for all $\tau \in \mathbb{T}_{m_t}$.
- Energy Harvesting: If $E(t) - \theta < 0$, perform energy harvesting and store the harvested energy during that frame, i.e., $e(\tau) = h(\tau)$ for all $\tau \in \mathbb{T}_{m_t}$. Else set $e(\tau) = 0$ for $\tau \in \mathbb{T}_{m_t}$.
- Queue Update: Update $Q^{(c)}(\tau)$ and $E(\tau)$ according to the dynamics (8) and (10), respectively. $\diamond$

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, $E(t) \le \theta + Th_{\max}$ 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, the node will always harvest energy when possible. In this case, one can introduce a virtual process $E'(t)$ to keep track of the energy level under OSA for decision making. It can be shown that this implementation gets a performance that is no worse than OSA.

We also 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 (19).

IV. Performance analysis

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 α is the maximum growth rate of the disutility. The parameter θ is defined to be:

$$\theta \triangleq V(\beta\delta + \frac{\beta|\mathcal{C}|R_{\max}}{P_{\min}} + \frac{\alpha d_{\max}}{P_{\min}}) + \delta T R_{\max} + T(P_{\max} + d_{\max}). \quad (23)$$

We 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 the harvestable energy process. As we will show later, (23) 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. Finally, note that the value θ depends on $\frac{R_{\max}}{P_{\min}}$ and $\frac{\alpha}{P_{\min}}$. This is because if the node enters the sleep mode to avoid energy outage (saving at least $TP_{\min}$ power), it will risk losing flow utility and suffer from disutility (at most $\beta|\mathcal{C}|R_{\max}+\alpha d_{\max}$). Hence, the energy storage capacity should be chosen to compensate the risk.

Theorem 1: Under the OSA algorithm with β and θ defined in (4) and (23), we have the following:

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

$$0 \le Q^{(c)}(t) \le \beta V + TR_{\max},\ \ \forall\ c, \quad (24)$$
$$0 \le E(t) \le \theta + Th_{\max}. \quad (25)$$

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

(b) Let $\overline{\boldsymbol{r}} = (\overline{r}^c, \forall\, c)$ and $\overline{D}$ be the time average admitted rate vector and time average disutility achieved by OSA. Then, we have:

$$\phi(\overline{\boldsymbol{r}}, \overline{D}) \ge \phi^* - \frac{B}{V}. \quad (26)$$

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

Here we present the proof for Part (a). The proof for Part (b) 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 (24). 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: (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 $O(1/\epsilon)$. This provides an explicit characterization of the size of the energy storage device needed for achieving the desired utility performance. Such explicit bounds are very useful in algorithm implementation.

V. Simulation

In this section, we simulate the OSA algorithm. We consider 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.,

$$D(b(t), d(t), \chi(m_t)) = \frac{1}{9}\chi(m_t)(d(t) - b(t))^2, \qquad (27)$$

which means $\alpha = 2/3$. The channel state $S(t)$ takes value in $\{0, 1\}$ equally likely. Then, when the node stays awake, we have $\mathcal{P}^{ON}_{\text{awake}} = \mathcal{P}^{OFF}_{\text{awake}} = \{1, 2, 3\}$. Thus, $P_{\min} = 1$ and $P_{\max} = 3$. $\mu(t) = \log(1 + 2S(t)(P(t) - P_{\min})$. 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 $U(r) = \log(1 + 2r)$, 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 OSA for $V \in \{10, 20, 40, 80, 100, 150, 200\}$. The results are plotted in Fig. 3. We see that as the V parameter increases, the average utility performance increases and quickly converges to the optimal. On the other hand, the average data queue size and the average energy level increases linear in V as per Theorem 1.

Fig. 4 shows a sample path energy level process under the OSA algorithm. We see that the energy level will never go 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 (25). 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.

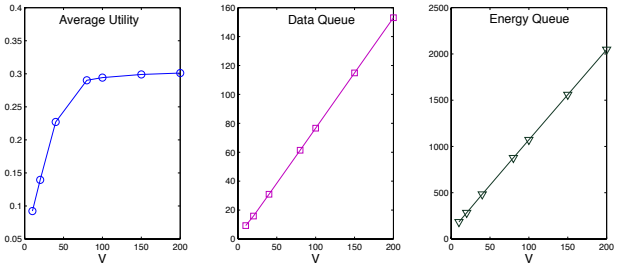

Fig. 3. Average utility, average data queue size and average energy queue size under OSA.

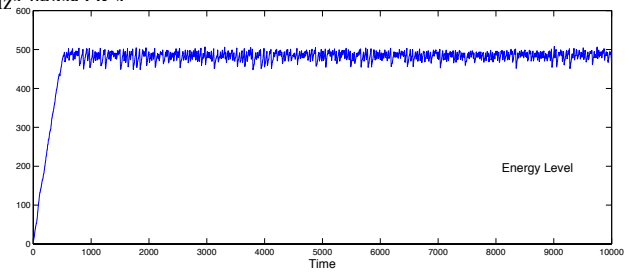

Fig. 4. A sample path energy level process from time 1 to 10000 under $V = 40$.

VI. Conclusion

In this paper, we develop the Optimal Sleep/wake scheduling Algorithm (OSA) for achieving optimal system utility for energy harvesting smart mobile devices powered by finite batteries. OSA is an online algorithm and does not require any knowledge of the harvestable energy processes. We show that OSA achieves an average utility that is within $O(\epsilon)$ of the optimal for any $\epsilon > 0$ using energy batteries of $O(1/\epsilon)$ sizes, while guaranteeing that the time average traffic congestion is $O(1/\epsilon)$.

VII. Acknowledgement

This work was supported in part by one or more of the following: the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61061130540, 61073174.

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:

$$\frac{1}{2}\sum_c \left([Q^{(c)}(\tau+1)]^2 - [Q^{(c)}(\tau)]^2\right) \qquad (28)$$
$$\leq \frac{1}{2}\sum_c \left([(R^{(c)}(\tau)]^2 + [\mu^{(c)}(\tau))]^2\right)$$
$$-1_{\text{w}}(m_\tau)\sum_c Q^{(c)}(\tau)\left[\mu^{(c)}(\tau) - R^{(c)}(\tau)\right].$$

Similarly, using (10), we have:

$$\frac{1}{2}\left([E(\tau+1) - \theta]^2 - [E(\tau) - \theta]^2\right) \qquad (29)$$
$$\leq \frac{1}{2}[b(\tau) + P(\tau)]^2 + \frac{1}{2}[e(\tau)]^2$$
$$-(E(\tau) - \theta)[1_{\text{w}}(m_\tau)(b(\tau) + P(\tau)) - e(\tau)].$$

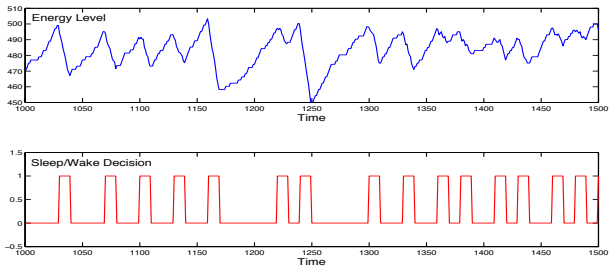

Fig. 5. A sample path energy level process and a sample path sleep/wake decision process under $V = 40$ from time 1000 to 1500.

Now define:

$$B_1 \triangleq \frac{1}{2}\Big[|\mathcal{C}|(R_{\max}^2+\mu_{\max}^2)+(d_{\max}+P_{\max})^2+h_{\max}^2\Big]. \quad (30)$$

Then, by summing (28) and (29), 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:

$$\Delta_T(t) \leq B_1 T \quad (31)$$
$$-\mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}\sum_{c} Q^{(c)}(\tau)1_{\rm w}(m_t)[\mu^{(c)}(\tau)-R^{(c)}(\tau)] \mid Z(t)\Big\}$$
$$-\mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1} E(\tau)\big[1_{\rm w}(m_t)[b(\tau)+P(\tau)]-e(\tau)\big] \mid Z(t)\Big\}.$$

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

$$Q^{(c)}(t_2) \leq Q^{(c)}(t_1)+(t_2-t_1)R_{\max},$$
$$Q^{(c)}(t_2) \geq Q^{(c)}(t_1)-(t_2-t_1)\mu_{\max},$$
$$E(t_2) \leq E(t_1)+(t_2-t_1)h_{\max},$$
$$E(t_2) \geq E(t_1)-(t_2-t_1)(d_{\max}+P_{\max}).$$

Using these inequalities in (31), we obtain that:

$$\Delta_T(t) \leq BT \quad (32)$$
$$-\mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}\sum_{c} Q^{(c)}(t)1_{\rm w}(m_t)[\mu^{(c)}(\tau)-R^{(c)}(\tau)] \mid Z(t)\Big\}$$
$$-\mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1} E(t)\big[1_{\rm w}(m_t)[b(\tau)+P(\tau)]-e(\tau)\big] \mid Z(t)\Big\}.$$

Here

$$B \triangleq B_1+\frac{|\mathcal{C}|(T-1)}{2}\Big[\mu_{\max}^2+R_{\max}^2 \quad (33)$$
$$+h_{\max}^2+(d_{\max}+P_{\max})^2\Big].$$

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

$$\Delta_T(t)-\sum_{\tau=t}^{t+T-1}\mathbb{E}\{Vf(\tau) \mid Z(t)\} \quad (34)$$
$$\leq BT-\sum_{\tau=t}^{t+T-1}\mathbb{E}\{Vf(\tau) \mid Z(t)\}$$
$$-\mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}\sum_{c} Q^{(c)}(t)1_{\rm w}(m_t)[\mu^{(c)}(\tau)-R^{(c)}(\tau)] \mid Z(t)\Big\}$$
$$-\mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1} E(t)\big[1_{\rm w}(m_t)[b(\tau)+P(\tau)]-e(\tau)\big] \mid Z(t)\Big\}.$$

Now using the definition 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 prove Part (b) of Theorem 1. We will use the following theorem, which states that there exists a stationary and randomized policy (and does not take into account the energy-availability constraint) that makes sleep/wake decisions, allocates power and achieves optimal utility.

***Theorem** 2:* [15] 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 packets 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 $t = mT, m = 0, 1, ...$

$$\mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1} f^{\Pi}(\tau)\Big\} = T\phi^*, \quad (35)$$

$$\mathbb{E}\Big\{1_{\rm w}^{\Pi}(m_t)\sum_{\tau=t}^{t+T-1}\big[\mu^{(c)\Pi}(\tau)-R^{(c)\Pi}(\tau)\big]\Big\} \geq 0, \quad (36)$$

$$\mathbb{E}\Big\{\sum_{\tau=t}^{t+T-1}\big[1_{\rm w}^{\Pi}(m_t)[b^{\Pi}(\tau)+P^{\Pi}(\tau)]-e^{\Pi}(\tau)\big]\Big\} = 0. \ \square \quad (37)$$

However, 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 [15]. We also note that the analysis here is different from the one in [11]. 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 allow us to compare our algorithm with alternative algorithms that choose 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 (19) for the given $E(t)$ and $Q^{(c)}(t)$. [8] Then, using (4), (5) and (24), we have:

$$D_{\rm tot}(m_t) \leq T|\mathcal{C}|V\beta R_{\max} \quad (38)$$
$$+\sum_{\tau=t}^{t+T-1}\Big[(V\beta+TR_{\max})\delta P^*(\tau)+(E(t)-\theta)P^*(\tau)\Big]$$

[8] Note that they may not be the implemented actions.

$$- \sum_{\tau=t}^{t+T-1} VD(0, d(\tau), \chi(m_t)) + \sum_{\tau=t}^{t+T-1} (E(t) - \theta) b^*(\tau)$$
$$+ \sum_{\tau=t}^{t+T-1} V\big[D(0, d(\tau), \chi(m_t)) - D(b(\tau)^*, d(\tau), \chi(m_t))\big].$$

Using the definition of θ in (23), 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 (38), we obtain:

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

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 of OSA that the RHS of the drift inequality (17) is indeed minimized under OSA, even over policies that do not consider the energy-availability constraint. Hence, the inequality remains valid if we plug in any alternative control policies that make two-stage decisions. In particular, we plug in the policy Π in Theorem 2 into (34) to have:

$$\Delta_T(t) - \sum_{\tau=t}^{t+T-1} \mathbb{E}\big\{Vf^{\text{OSA}}(\tau) \mid Z(t)\big\} \leq BT - VT\phi^*. \quad (39)$$

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

$$\mathbb{E}\big\{L(KT) - L(0)\big\} - \sum_{\tau=0}^{KT-1} \mathbb{E}\big\{Vf^{\text{OSA}}(\tau)\big\} \leq BKT - VKT\phi^*. \quad (40)$$

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

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

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

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

Using Jensen's inequality, we conclude that:

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

This completes the proof of Part (b). ■

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 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. *Proc. 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] 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.

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

[9] 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.

[10] 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.

[11] L. Huang and M. J. Neely. Utility optimal scheduling in energy harvesting networks. *Proceedings of ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, May 2011.

[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] W. Lai and I. C. Paschalidis. Optimally balancing energy consumption versus latency in sensor network routing. *ACM Transactions on Sensor Networks, Vol. 4, No. 4, Article 21*, August 2008.

[14] 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.

[15] 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.

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

[17] M. J. Neely and L. Huang. Dynamic product assembly and inventory control for maximum profit. *IEEE Conference on Decision and Control (CDC), Atlanta, Georgia*, Dec. 2010.

[18] L. Huang, J. Walrand, and K. Ramchandran. Optimal demand response with energy storage management. *Proceedings of IEEE International Conference on Smart Grid Communications (SmartGridComm)*, November 2012.

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

[20] L. Huang and M. J. Neely. Delay reduction via Lagrange multipliers in stochastic network optimization. *IEEE Transactions on Automatic Control, Volume 56, Issue 4, pp. 842-857*, April 2011.