Energy Efficient Sleep/Wake Scheduling for Multi-hop Sensor Networks: Non-convexity and Approximation Algorithm

Yan Wu, Sonia Fahmy, Ness B. Shroff
Center for Wireless Systems and Applications (CWSA), Purdue University

***Abstract*— We study sleep/wake scheduling for low duty cycle sensor networks. Our work is different from prior work in that we explicitly consider the effect of synchronization error in the design of the sleep/wake scheduling algorithm. In our previous work, we have studied sleep/wake scheduling for *single hop* communications, e.g., intra-cluster communications between a cluster head and cluster members. We showed that the there is an inherent trade-off between energy consumption and message delivery performance (defined as the message capture probability). We proposed an optimal sleep/wake scheduling algorithm, which satisfies a message capture probability threshold (assumed to be *given*) with minimum energy consumption.**

In this work, we consider multi-hop communications. We remove the previous assumption that the capture probability threshold is *already given*, and study how to decide the per-hop capture probability thresholds to meet the Quality of Services (QoS) requirements of the application. In many sensor network applications, the QoS is decided by the amount of data delivered to the base station(s), i.e., the multi-hop delivery performance. We formulate an optimization problem, which aims to set the capture probability threshold at each hop such that the network lifetime is maximized, while the multi-hop delivery performance is guaranteed. The problem turns out to be non-convex and hard to solve exactly. By investigating the unique structure of the problem and using approximation techniques, we obtain a solution that achieves at least 0.73 of the optimal performance.

I. Introduction

An important class of wireless sensor network applications is the class of continuous monitoring applications. These applications employ a large number of sensor nodes for continuous sensing and data gathering. Each sensor *periodically* produces a small amount of data and reports to one (or several) base station(s). This application class includes many typical sensor network applications such as habitat monitoring [1] and civil structure monitoring [2].

Measurements show that idle listening consumes a significant amount of energy for sensor devices. An effective approach to conserve energy is to put the radio to sleep during idle times and wake it up right before message transmission/reception. This requires precise synchronization between the sender and the receiver, so that they can wake up simultaneously to communicate with each other. The state-of-the-art in sleep/wake scheduling assumes that the underlying synchronization protocol can provide nearly perfect (e.g., μs level) synchronization, so that clock disagreement can be ignored. However, in our previous work [3], we determined that the impact of synchronization error is non-negligible. We found that although existing synchronization schemes achieve precise synchronization *immediately after* the exchange of synchronization messages, there is still random synchronization error because of the non-deterministic factors in the system. Due to the synchronization error, clock disagreement grows with time and can be comparable to the actual message transmission time. This means the design of an effective sleep/wake scheduling algorithm must consider the impact of synchronization error. We showed that there is an inherent trade-off between energy consumption and message delivery performance (defined as the message capture probability). We then proposed an optimal sleep/wake scheduling algorithm, which achieves a message capture probability threshold (assumed to be *given*) with minimum energy consumption.

In the above-mentioned work, we focused on single-hop communications. In this work, we consider *multi-hop* communications. For illustration, we consider a network that has been *hierarchically clustered*. We remove the previous assumption that the capture probability threshold is *already given*, and study how to decide the per-hop capture probability thresholds to meet the QoS requirement of the application. In many sensor network applications, the nodes cllect sensing data and report to the base station(s) (BS). Therefore, the QoS is decided by the amount of data delivered from the nodes to the BS, i.e., the multi-hop delivery performance. We formulate an optimization problem which aims to set the capture probability threshold at each hop such that the network lifetime is maximized, while a minimum fraction of data is guaranteed to be delivered to the BS. The problem turns out to be non-convex and hard to solve exactly. Therefore, we use approximation techniques and obtain a 0.73-approximation algorithm.

The remainder of this paper is organized as follows. Section II reviews related work. Section III gives the system model and briefly describes our sleep/wake scheduling algorithm for single hop communications. Section IV studies how to assign the thresholds along multi-hop paths in the cluster hierarchy. Section V concludes the paper.

– Yan Wu and Sonia Fahmy are also with the Department of Computer Science, Purdue University, West Lafayette, IN 47907–2107, USA. E-mail: {wu26, fahmy}@cs.purdue.edu. Ness B. Shroff is also with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907–2035, USA. E-mail: shroff@ecn.purdue.edu.

– This research has been sponsored in part by NSF grants 0238294 and 0207728, an Indiana 21st century grant, and a Tellabs foundation fellowship.

II. Background and Related Work

We first review previous work on sleep/wake scheduling in sensor networks, then discuss clustering in sensor networks.

A. Sleep/Wake Scheduling for Sensor Networks

MAC designs for wireless sensor networks can be broadly classified as contention-based or TDMA protocols. In contention-based MACs, overhearing, collisions, and idle listening waste energy. Hence, mechanisms such as overhearing avoidance have been proposed to save energy with contention-based MACs [4]–[6]. Several researchers argue that TDMA protocols combined with sleep/wake scheduling are more suited to sensor network applications (since TDMA protocols avoid energy waste due to contention). In this case, the radio sleeps during idle times, and wakes up right before message transmission/reception. This requires precise synchronization between the sender and the receiver, so that they can wake up at the same time to communicate with each other. Most existing sleep/wake scheduling schemes assume that the underlying synchronization protocol can provide nearly perfect (e.g., μs level) synchronization, and that clock disagreement is negligible at all times. This, however, is untrue in practice. Consider two nodes that have agreed to rendezvous on the radio channel once every 5 minutes to exchange a 30-byte message. Using a 19.2 kbps radio such as RF Monolithics [7], 30 bytes can be transmitted in 12.5 ms. The radio must be awakened early to compensate for clock disagreement. Let the relative clock skew be 5 ppm[1], i.e., the clocks of the two nodes drift away from each other 5 μs each second. After 5 minutes, the clocks will drift by $5\ \mu s \times 300 = 1.5\ ms$, which is non-negligible compared to the message transmission time.

B. Clustering for Sensor Networks

Clustering is generally considered to be a scalable method to manage large sensor networks. Sensors within a geographical region are grouped into a cluster. The sensors are then locally managed by a cluster head (CH) – a node elected to coordinate the nodes within the cluster and to be responsible for communication between the cluster and the BS or other cluster heads. This grouping process can be recursively applied to build a cluster hierarchy. Sensor nodes first elect level-1 CHs, then level-1 CHs elect a subset of themselves as level-2 CHs. Cluster heads at levels $3, 4, \ldots$ are elected in a similar fashion to generate a hierarchy of CHs, in which any level-i CH is also a CH of level $(i-1), (i-2), \ldots, 1$. Fig. 1 depicts nodes organized in a three-level cluster hierarchy with each number representing the level of the corresponding node.

Hierarchical clustering provides a convenient framework for resource management and local decision making. Moreover, it can be extremely effective for data fusion, i.e., sensing data can be aggregated before being passed onto the next higher level in the hierarchy. Hence, hierarchical clustering is used in many practical systems [2], [9]. Due to this widespread use, in this work we choose the cluster hierarchy model as an illustrative example. We assume that the network has been hierarchically clustered using one of the popular clustering techniques [10], [11].

[1]According to the datasheet of Mica Motes [8], the clock skew with respect to the standard clock is up to 50 ppm, thus the relative clock skew between two sensor nodes can be 100 ppm in the worst case.

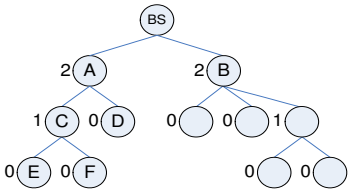

Fig. 1. A three-level cluster hierarchy

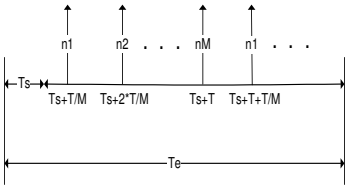

Fig. 2. Equispaced upstream transmissions

III. System Model

We consider a cluster hierarchy, where each cluster consists of a single cluster head (CH) and multiple cluster members. Note that a node can be both the CH in one cluster, and a member in another cluster at a higher level, e.g., in Fig. 1, C is the CH of E, but is also a member of A. Time is divided into recurring *epochs* with constant duration T_e. As in many MAC protocols for sensor networks [5], [6], each epoch begins with a synchronization interval T_s followed by a transmission interval (Fig. 2). During the synchronization interval, the cluster members synchronize with their CH and no transmissions are allowed. During the transmission interval, each member node transmits in a TDMA manner and sends one message to the CH every T seconds. The message consists of the aggregate of its own sensing data, and the data collected from its members if the node itself is a CH. Each transmission interval contains one or more rounds of transmissions, i.e., $T_e = T_s + NT, N \geq 1$. The transmissions from the different members are equispaced, i.e., if M is the number of cluster members, then transmissions are separated by $\frac{T}{M}$.

A. Assumptions

We make the following assumptions about our system:

(1) Orthogonal Frequency Channels: We assume that neighboring clusters use orthogonal frequency bands and do not interfere with each other. This is a reasonable assumption since the data rate of sensor networks is usually low, typically around 10−40 kbps. If we run the network in ISM-900 bands (902−928 MHz), then there are more than a thousand frequency channels to choose from.

A node that is both a CH and cluster member needs to communicate with its members and with its CH, e.g., in Fig. 1 node C needs to communicate with both A and E. However, A and E are in neighboring clusters; hence they use different frequency channels. Since every node has only one radio interface, C has to schedule carefully to participate in each cluster. This can be achieved in the following manner. The BS first decides the schedule of the synchronization interval and the transmission schedule for its members (A and B in Fig. 1), then broadcasts this information to the members. A and B, upon hearing the broadcast, will reserve the relevant times for synchronizing/communicating with the root. Then, A

and B schedule the synchronization and transmissions for their members at different times. Similarly, C will reserve the times to synchronize/communicate with A, and choose different times for its members (E and F) to synchronize/transmit.

(2) Data aggregation: We adopt a data aggregation model similar to [12]. Consider a cluster with node 0 being the CH, and with M members, $i = 1, \ldots, M$. The length of messages from node i is L_i, $i = 0, \ldots, M$. Thus, the length of the aggregated message is a function of $L_i, i = 0, \ldots, M$. We use the following model for $\chi(L_0, \ldots L_M)$, the length of the aggregated message,

$$\chi(L_0, \ldots L_M) = r \sum_{i=0}^{M} L_i + c. \quad (1)$$

In this model, c corresponds to the overhead of aggregation, while $r \leq 1$ is the compression ratio. Note that r can be 0, in which case Eq. (1) corresponds to the case when all messages can be combined into a single message of fixed length. This models those applications where we want updates of type min, max, and sum (e.g., event count).

The model in Eq. (1) assumes the same compression ratio for messages from different nodes. It can be easily extended to account for different compression ratios, and the results still apply. The details can be found in our technical report [13].

(3) Radio hardware: We assume that the sender can precisely control when the message is sent out onto the channel using its *own* clock. This is reasonable since in [14], system measurements have shown that non-determinism at the sender is negligible compared to non-determinism at the receiver.

For the receiver, we assume that if there is an incoming message, it can immediately detect the radio signal. This is a close approximation of the real situation, since modern transceivers can detect incoming signals within microseconds [15]. Further, we assume that once the receiver detects an incoming message, it will stay active until the reception is completed.

(4) Collisions: We assume that the separation between transmissions from different members, $\frac{T}{M}$, for a cluster with M members is large enough so that the collision probability for transmissions from different members is negligible. This is a reasonable assumption for low duty cycle sensor networks. Consider a large cluster of $M = 50$ members and each member transmits to the CH every $T = 60$ seconds. The separation is $\frac{T}{M} = 1200\ ms$. For low duty cycle networks, the message size is usually not large; hence the transmission time is much smaller than this separation. Moreover, at the beginning of each epoch, the cluster members re-synchronize with the CH, so that the clock disagreement will not become large enough to cause significant collision probability.

(5) Propagation delay: Finally, because the communication range for sensor nodes is typically < 100 meters, the propagation delay is below $1\mu s$. Thus, we consider the propagation delay to be negligible and assume it to be zero for simplicity.

B. Synchronization Algorithm

Time synchronization for wireless sensor networks has been extensively investigated [14], [16]–[20]. Clock disagreement between sensor nodes can be characterized using two factors: *phase offset* and *clock skew*. Phase offset corresponds to clock disagreement between nodes at a given instant. Clock skew means clocks run at different speeds, i.e., the actual frequency deviates from the expected frequency. This is due to manufacturing imprecision and aging effects. The maximum clock skew is less than than 100 ppm and is usually specified by the manufacturer. Besides manufacturing imprecision and aging, the frequency is also affected by environmental factors including temperature, pressure, and voltage [21]. Among these factors, temperature has the most significant effect. When temperature significantly changes, the variation in the clock frequency can be up to several tens of ppm, while the variation caused by other factors is far below 1 ppm. Observe, however, that temperature does not change dramatically within a few seconds in typical sensor environments. If the epoch duration T_e is chosen according to the temperature change properties of the environment, we can assume that the clock skew for each node is constant over each epoch. This is consistent with the empirical observations in [20].

In this work, we adopt the well-known RBS synchronization scheme, and study the sleep/wake scheduling problem[2]. The scheme includes two steps: (1) Exchange synchronization messages to obtain multiple pairs of corresponding time instants; and (2) Use linear regression to estimate the clock skew and phase offset.

At the beginning of each epoch j, the members need to synchronize with the CH. To this end, each member i exchanges several synchronization messages with the CH and obtains N_s pairs of corresponding time instants $(C(j,k), t_i(j,k)), k = 1, \ldots, N_s$, where $C(j,k), t_i(j,k)$ denote the k^{th} time instant of the CH and of node i in epoch j respectively.

Under the assumption that the clock skew of each node does not change over the epoch, during a given epoch j the clock time of member node i, t_i, is a linear function of the CH clock time C, i.e., $t_i(C) = a_i(j)C + b_i(j)$, where $a_i(j), b_i(j)$ denote the relative clock skew and phase offset (respectively) between member node i and CH in epoch j.

Because of the non-determinism in the message exchange, the obtained time correspondence is not exactly accurate and contains an error, i.e.,

$$t_i(j,k) = a_i(j)C(j,k) + b_i(j) + e_i(j,k), \quad (2)$$

where $e_i(j,k)$ is the random error caused by non-determinism in the system. Real system measurements [16] show with a high confidence level that $e_i(j,k)$ follows a well-behaved *normal* distribution with zero mean $N(0, \sigma_0^2)$, and σ_0 is on the order of several tens of microseconds.

At each epoch j, pairs $(C(j,k), t_i(j,k)), k = 1, \ldots, N_s$ are obtained via exchange of synchronization messages. Then, linear regression is performed on these N_s pairs to obtain estimates of $a_i(j), b_i(j)$, denoted by $\hat{a}_i(j), \hat{b}_i(j)$. In this work, we control the exchange of synchronization messages such that $C(j,k) \approx jT_e + k\frac{T_s}{N_s}, j = 1, \ldots, \infty, k = 1, \ldots, N_s$. This can be achieved by letting the CH initiate the message exchange, and we give the detailed explanation in our technical report [13].

[2]This scheme is chosen for illustration purposes only. Our sleep/wake scheduling solution works with most synchronization schemes.

C. The Optimal Sleep/Wake Scheduling Problem

We now summarize our previous work [3] on sleep/wake scheduling for single hop intra-cluster communications.

Assume that during epoch j, node i has a message p to send at CH clock time τ_p, where $jT_e \leq \tau_p \leq (j+1)T_e$. Node i first converts τ_p into its own clock time using the estimates $(\hat{a}_i(j), \hat{b}_i(j))$, i.e., $\hat{t}_i(\tau_p) = \hat{a}_i(j)\tau_p + \hat{b}_i(j)$, and then it sends out the message at $\hat{t}_i(\tau_p)$ according to its own clock.

The CH clock time corresponding to $\hat{t}_i(\tau_p)$ is:

$$\tau'_p = \frac{\hat{t}_i(\tau_p) - b_i(j)}{a_i(j)} = \tau_p + \frac{(\hat{a}_i(j) - a_i(j))\tau_p + \hat{b}_i(j) - b_i(j)}{a_i(j)}. \quad (3)$$

From Eq. (2), random errors exist in the measurements. Therefore, $(\hat{a}_i(j), \hat{b}_i(j))$ is also random and may not equal $(a_i(j), b_i(j))$. As a result, the actual arrival time τ'_p will deviate from the scheduled arrival time τ_p. To account for this random deviation and still "capture" (receive) the message, the CH needs to wake up earlier than τ_p and stay active for some time (Fig. 3), i.e., it uses a wake up interval to capture the actual message arrival. This leads to the following question: *When should the CH wake up and how long should the wake up interval be?*

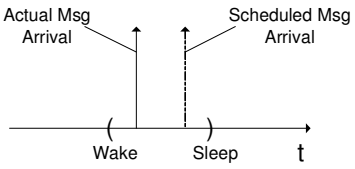

Fig. 3. Wake up interval to capture the message

Obviously, if the CH wakes up much earlier than τ_p and stays active for a long time, the probability of capturing the message would be high; however, waking up early and staying active for a long time consumes more energy. In order to reduce energy consumption, yet still guarantee the message delivery performance, we formulate the following optimal sleep/wake scheduling problem which attempts to minimize the expected energy consumption with constraints on the capture probability.

Let p be a message from node i to arrive in epoch j, i.e., scheduled arrival time $\tau_p \in (jT_e, jT_e + T_e)$. Let τ'_p be the actual arrival time at which p arrives at the CH, as defined in Eq. (3). To capture p, the CH wakes up at w_p, and waits for the message until s_p. The goal is to determine w_p and s_p to minimize the expected energy consumption as described by the following optimization problem:

(A) Min $E = (s_p - w_p)\alpha_I Prob\{\tau'_p \notin (w_p, s_p)\} + \int_{w_p}^{s_p} \{(x - w_p)\alpha_I + \frac{L_p}{R}\alpha_r\} f_{\tau'_p}(x)dx$
such that $Prob\{\tau'_p \in (w_p, s_p)\} \geq th$,

where α_I/α_r is the power consumption during idle/reception time; L_p is the length of the message; R is the data rate; $f_{\tau'_p}(\cdot)$ is the probability density function (PDF) of τ'_p; and th is the capture probability threshold, $0 < th < 1$. Its value is application specific and is assumed to be *given*.

To solve (A), we first compute the PDF $f_{\tau'_p}(x)$. By linear regression [22], we find that τ'_p is *normally* distributed and

$$E(\tau'_p) = \tau_p, \quad (4)$$

$$VAR(\tau'_p) \equiv \sigma_p^2 = \frac{\sigma_0^2}{a_i^2(j)} \frac{1}{N_s} [1 + \frac{(\tau_p - \overline{C(j,k)})^2}{\overline{C^2(j,k)} - (\overline{C(j,k)})^2}],$$

where $\overline{C(j,k)} = \frac{\sum_{k=1}^{N_s} C(j,k)}{N_s}$, $\overline{C^2(j,k)} = \frac{\sum_{k=1}^{N_s} C^2(j,k)}{N_s}$.

Substituting Eq. (4) into problem (A), and letting $\hat{\tau} = \frac{\tau'_p - \tau_p}{\sigma_p}$, $w = \frac{w_p - \tau_p}{\sigma_p}$, $s = \frac{s_p - \tau_p}{\sigma_p}$, i.e., $\hat{\tau}, (w, s)$ are the normalized arrival time and normalized wake up interval respectively. After simple algebraic operations, problem (A) becomes:

(A1) Min $F(w,s) = (s-w)\sigma_p\alpha_I - [Q(w) - Q(s)]s\sigma_p\alpha_I + [g(w) - g(s)]\sigma_p\alpha_I + [Q(w) - Q(s)]\frac{L_p}{R}\alpha_r$,
such that $Q(w) - Q(s) \geq th$,

where $g(\cdot)$ is the PDF for the standard normal distribution, and $Q(\cdot)$ is the complementary cumulative distribution function.

The main difficulty in solving (A1) is that the problem is not a convex optimization problem, which can be shown by computing the Hessian matrix. Due to the non-convexity, we cannot use conventional convex optimization techniques [23] to find the optimal solution. Hence, we looked into the structure of problem (A1) and proved the following proposition, which shows that the optimal solution always appears at the boundary of the region $Q(w) - Q(s) \geq th$.

Proposition 1: For Problem (A1), any optimal solution (w^*, s^*) satisfies $Q(w^*) - Q(s^*) = th$.

The equality $Q(w^*) - Q(s^*) = th$ means that under the optimal scheduling policy, the capture probability is always equal to the threshold th. This is quite intuitive as we can imagine that to guarantee a larger capture probability, more energy will be consumed.

Substituting $Q(w^*) - Q(s^*) = th$ into $F(w,s)$, we get

$$F(w,s) = [(1-th)s - w + g(w) - g(s)]\sigma_p\alpha_I + th\frac{L_p}{R}\alpha_r. \quad (5)$$

Substituting Eq. (5) into (A1), we can simplify the formulation as follows. First, because $th\frac{L_p}{R}\alpha_r$ does not depend on w and s, we remove it from $F(w,s)$. Second, all the remaining terms of $F(w,s)$ have $\sigma_p\alpha_I$, so we can extract $\sigma_p\alpha_I$. Finally, because $Q(x)$ is monotonic, we express s as a function of w, $s(w) = Q^{-1}(Q(w) - th)$. Now the formulation becomes:

(A2) Min $G(w) = (1-th)s(w) - w + g(w) - g(s(w))$,
such that $s(w) = Q^{-1}(Q(w) - th)$ and $w < Q^{-1}(th)$.

Hence, we have transformed the original formulation (A) into an equivalent formulation (A2). We can see that the minimum expected energy to receive the message is

$$\sigma_p\alpha_I\gamma(th) + \frac{L_p}{R}\alpha_r th, \quad (6)$$

where

$$\gamma(th) = \min\{G(w) : w < Q^{-1}(th)\} \quad (7)$$

is the minimum value of the objective function in (A2). Eq. (6) and (7) will be used later in Section IV.

Next, we solve (A2). The following proposition shows that $G(\cdot)$ is a convex function, and gives the position of the global minimum.

Proposition 2: (1) $G^{''}(w) > 0$;

(2) Let w_0 be the global minimum, $w_l = Q^{-1}(\frac{1+th}{2}), w_u = \min(0, Q^{-1}(th))$, then $w_0 \in (w_l, w_u)$, and is the unique minimum on this interval.

Because w_0 is the unique minimum on (w_l, w_u), we can use the Golden Search method to find w_0 [24]. The complexity of the Golden Search method is $O(log(\frac{1}{\delta}))$, where δ is the required precision. Thus, it can be efficiently implemented.

After we obtain w^*, s^*, we compute the optimal sleep/wake schedule as $(w_p^* = \tau_p + w^*\sigma_p, s_p^* = \tau_p + s^*\sigma_p)$.

We simulated our sleep/wake scheduling scheme and showed that it outperforms schemes that do not intelligently consider the synchronization error [13].

IV. The Capture Probability Threshold Assignment Problem

As described in Section III-C, our previous work [3] focused on single hop intra-cluster communications, and studied the optimal sleep/wake scheduling problem under the assumption that the capture probability threshold was *already given*. In this work, we study how to decide the capture probability threshold to meet the QoS requirement of the application and maximize the network lifetime.

A. Problem Definition

Consider a sensor network deployed for environment monitoring. The network consists of a set of sensor nodes denoted by S, and one or more base stations (BSs), usually personal computers. The network has already been hierarchically clustered using one of these clustering techniques [10], [11]. We assume there is a single BS, denoted by BS. The formulation can be easily extended to the case with multiple BSs. $H(n)$ denotes the cluster head of node n. $M(n)$ denotes the set of nodes that are members of n. $D(n)$ denotes the set of nodes that are the descendants of n. $M(n)$ and $D(n)$ can be empty if node n is at level 0. $d(n)$ is the hop distance from node n to BS, i.e., $H^{(d(n))}(n) \equiv \underbrace{H(H(\ldots H(n)\ldots))}_{d(n)} = BS$.

Each sensor node periodically reports to its CH. The CH aggregates its own sensing data and the data collected from the members over the last transmission period, then forwards the aggregated data to its CH. The process continues until the message finally gets to BS. Each message contains some sensing data and represents certain amount of "information" about the environment. BS uses the collected information to compute certain properties, e.g., the chemical contaminant in the area. The service quality is defined as the accuracy of the computed properties, which is decided by the amount of information collected by BS, i.e., the more information collected, the better accuracy. Hence, the service quality is not decided by the delivery performance at any particular hop, but by the *multi-hop delivery performance* from the nodes to BS.

However, collecting more information requires higher energy consumption and may lead to widely varying power dissipation levels across nodes, e.g., nodes at high levels in the cluster hierarchy have an excessive relaying burden. This will result in a shorter lifetime for some nodes, which can lead to loss of coverage when these nodes deplete their energy. This is the inherent trade-off between application performance and network lifetime. To maximize the network lifetime and still guarantee the application performance, we formulate the following optimization problem.

We define the network lifetime T_L as the time until the death of the first sensor node. This definition is widely used in the literature [4], [10], [25]–[27]. It mainly applies to application scenarios with strict coverage requirements, where each sensor "covers" a certain area in the environment and provides equally important information to BS. To maintain complete coverage and save redeployment cost, we must ensure that all the nodes remain up for as long as possible[3].

Let $z(n)$ be the capture probability threshold of $H(n)$ for messages coming from n, i.e., node $H(n)$ will capture messages from node n with probability no less than $z(n)$. The goal is to choose $z(n)$ to maximize the network lifetime, and still guarantee that all information be delivered to BS with a predefined probability Λ:

(B) Max T_L
such that $\prod_{i=0}^{d(n)-1} z(H^{(i)}(n)) \geq \Lambda, \forall n \in S$,

where Λ is decided by the QoS requirement of the application.

For the data from node n to be received by BS, it needs to pass through $H(n), H^{(2)}(n), \ldots, H^{(d(n)-1)}(n)$. Hence in (B), the constraint $\prod_{i=0}^{d(n)-1} z(H^{(i)}(n)) \geq \Lambda$ means the data from n will be received by BS with probability no less than Λ. Note that the data will be aggregated with data from other nodes at each hop along the path.

B. Solution

In the cluster hierarchy, if the multi-hop delivery performance of a leaf node (a level-0 node) is guaranteed, then the delivery performance for its ancestors is guaranteed as well, i.e., if the information from a leaf node n is delivered to BS with probability no less than Λ, then the information from $H(n), H^{(2)}(n) \ldots H^{(d(n)-1)}(n)$ will also be delivered with probability no less than Λ. Hence, in (B), the constraints on the delivery performance of non-leaf nodes are redundant and can be removed. Let LF denote the set of leaf nodes. We obtain the following formulation:

(B) Max T_L
such that $\prod_{i=0}^{d(n)-1} z(H^{(i)}(n)) \geq \Lambda, \forall n \in LF$,

To obtain an explicit form of Problem (B), we characterize the average power dissipation for each sensor node when $z(m), m \in S$ are given. During an epoch, a node n consumes energy for sensing, synchronization, and transmitting/receiving data messages. Let the sensing energy and synchronization energy be $\varepsilon_s(n)$, and $\varepsilon_{syn}(n)$ respectively. These do not depend on the capture probability thresholds.

Both the transmission energy and the receiving energy depend on the capture probability thresholds. Let l be the

[3]Here, we assume that we will lose the corresponding coverage if a node dies, i.e., there is no redundant node. If the network has redundancy, we can consider the nodes covering the same area (e.g., nodes near the same bird nest) as a single node whose initial energy equals the sum of energy of all the relevant nodes, and then this definition and the following results still apply.

amount of sensing data generated by each sensor during each transmission period T, and $L^{avg}(n)$ be the *average* message size from n. Then, from the aggregation model in Eq. (1),

$L^{avg}(n) = r(l + \sum_{i \in M(n)} z(i) L^{avg}(i)) + c.$

Recursively applying the above formula, we have

$$L^{avg}(n) = rl + c + \sum_{i \in D(n)} (rl + c) \prod_{k=0}^{d(i)-d(n)-1} [rz(H^{(k)}(i))]. \quad (8)$$

Since N messages are transmitted in each epoch, the average transmission energy in an epoch is

$$\varepsilon_t(n) = N\alpha_t(n)\frac{L^{avg}(n)}{R}, \quad (9)$$

where $\alpha_t(n)$ is the transmission power of node n[4].

We now compute the average receiving energy $\varepsilon_r(n)$. For a node n with $|M(n)|$ members, during a given epoch j, these nodes transmit to n in turn. To decide the transmission sequence, node n orders the $|M(n)|$ members, i.e., each member node $i \in M(n)$ is assigned a sequence number $\theta(i)$ from $\{1, 2, \ldots, |M(n)|\}$, and different member nodes have different sequence numbers. Node i is scheduled to transmit at $jT_e + T_s + \theta(i)\frac{T}{|M(n)|} + hT, h = 1, \ldots, N$. For given capture probability thresholds, node n will use the sleep/wake schedule described in Section III-C, as it is the *optimal sleep/wake schedule*. Therefore, the average energy used to receive a message scheduled to arrive at τ_p is exactly the minimum value of the objective function in Problem (A), which is (by Eq. (6))

$$\sigma_p \alpha_I \gamma(th) + \frac{L_p}{R}\alpha_r th.$$

Here, L_p is the message size, σ_p is computed from Eq. (4), th is the required threshold, and $\gamma(th)$ is as given in Eq. (7). The average receiving energy $\varepsilon_r(n)$ can be computed by summing up the energy used to receive all messages from its members. As in Section III-B, the synchronization is controlled such that $C(j,k) \approx jT_e + k\frac{T_s}{N_s}$, so

$$\begin{aligned} \overline{C(j,k)} &\approx jT_e + \frac{1+N_s}{2}\frac{T_s}{N_s}, \quad (10) \\ \overline{C^2(j,k)} - (\overline{C(j,k)})^2 &\approx \frac{\sum_{k=1}^{N_s}(k\frac{T_s}{N_s} - \frac{1+N_s}{2N_s}T_s)^2}{N_s}. \end{aligned}$$

Further, recall that the maximum clock skew is no larger than 100 ppm; hence in Eq. (4), the relative clock skew $a_i(j) \approx 1$. Combining these together, we have

$$\varepsilon_r(n) \approx \sum_{i \in M(n)} \sum_{h=1}^{N} \alpha_r z(i)\frac{L^{avg}(i)}{R} + \quad (11)$$
$$\alpha_I \gamma(z(i)) \sqrt{\sigma_0^2 \frac{1}{N_s}[1 + \frac{(T_s + \frac{\theta(i)T}{|M(n)|} + hT - \frac{1+N_s}{2}\frac{T_s}{N_s})^2}{\frac{\sum_{k=1}^{N_s}(k\frac{T_s}{N_s} - \frac{1+N_s}{2N_s}T_s)^2}{N_s}}]}.$$

For node n, the average energy consumption in an epoch is the sum of the sensing energy, the synchronization energy, and the transmission/reception energy. Combining Eq. (8), (9) and (11), the average power dissipated in node n is given by

$$\begin{aligned} \eta(n, \overrightarrow{z}) &= \frac{\varepsilon_s(n) + \varepsilon_{syn}(n) + \varepsilon_t(n) + \varepsilon_r(n)}{T_e} \quad (12) \\ &= A(n) + \sum_{i \in M(n)} P(n,i)\gamma(z(i)) + \\ &\quad \sum_{i \in D(n)} Q(n,i) \prod_{k=0}^{d(i)-d(n)-1} z(H^{(k)}(i)), \end{aligned}$$

where

$$A(n) = \frac{1}{T_e}[\varepsilon_s(n) + \varepsilon_{syn}(n) + N\alpha_t(n)\frac{rl+c}{R}],$$

$$P(n,i) = \frac{1}{T_e}\sum_{h=1}^{N}\alpha_I \sqrt{\sigma_0^2 \frac{1}{N_s}[1 + \frac{(T_s + \frac{\theta(i)T}{|M(n)|} + hT - \frac{1+N_s}{2}\frac{T_s}{N_s})^2}{\frac{\sum_{k=1}^{N_s}(k\frac{T_s}{N_s} - \frac{1+N_s}{2N_s}T_s)^2}{N_s}}]},$$

$$Q(n,i) = \frac{1}{T_e}\frac{rN\alpha_t(n) + N\alpha_r}{R}(rl + c)r^{d(i)-d(n)-1}.$$

Let $\xi(n)$ be the initial energy of node n, then Problem (B) can be written as

Max T_L
such that $\prod_{i=0}^{d(n)-1} z(H^{(i)}(n)) \geq \Lambda, \forall n \in LF$,
$\eta(n, \overrightarrow{z}) \leq \xi(n)/T_L, \forall n \in S$.

Next, we introduce a lifetime-penalty function $\Psi(1/T_L)$ to be a strictly convex and increasing function (e.g., $\Psi(x) = x^2$). Then, maximizing the network lifetime is equivalent to minimizing the lifetime-penalty function. We now use a change of variable $u = 1/T_L$ to give the network lifetime maximization problem as the following equivalent problem:

(B) Min $\Psi(u)$
such that $\prod_{i=0}^{d(n)-1} z(H^{(i)}(n)) \geq \Lambda, \forall n \in LF$,
$\eta(n, \overrightarrow{z}) \leq \xi(n)u, \forall n \in S$.

The difficulty in solving (B) is that it is not a convex optimization problem. To see this, we observe that in the second set of constraints, the left side $\eta(n, \overrightarrow{z})$ includes $\gamma(z(i))$ and $\prod z(i)$. $\prod z(i)$ may not be convex, e.g., $z(1)z(2)$; for $\gamma(z(i))$, we numerically show the curve in Fig. 4 which is clearly not convex. Hence, the constraint region is not a convex set, and Problem (B) is not convex. Further, we do not have an explicit analytical form for $\gamma(z)$. This makes Problem (B) hard to solve exactly. Next, we investigate the structure of the problem and obtain an approximate solution.

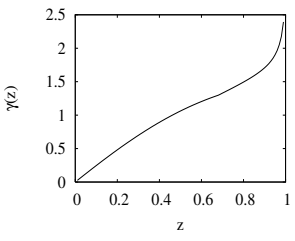

Fig. 4. $\gamma(z)$

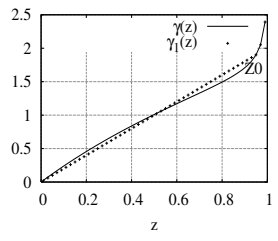

Fig. 5. Approximating $\gamma(z)$

The following proposition characterizes $\gamma(z)$.

[4]We assume that each node has a fixed number of transmission power levels (as in Mica2 motes), and can choose the appropriate one based upon factors such as distance and channel fading.

Proposition 3: (1) For $z \geq 0.86$, $\gamma(z)$ is strictly convex;
(2) For $z \in [0, 0.99], 1.86z < \gamma(z) < 2.52z$.

We give the proof in the technical report [13]. The idea is that although we do not have an explicit analytical form of $\gamma(z)$, we have the bounds obtained from Proposition 2(2). Therefore, we compute $\gamma'(z), \gamma''(z)$ using implicit differentiation and bound them. This proposition shows that $\gamma(z)$ is convex in the region $[0.86, 1)$; for the remaining region where $\gamma(z)$ may not be convex, we can bound it fairly tightly.

Next, we approximate $\gamma(z)$ with a convex function. The curve $2z + 0.001z^2$ intersects $\gamma(z)$ at $Z_0 \approx 0.95$. Let

$$\gamma_1(z) = \begin{cases} 2z + 0.001z^2 & 0 \leq z \leq Z_0 \\ \gamma(z) & Z_0 \leq z < 1 \end{cases}$$

The following proposition shows that $\gamma_1(z)$ is a *convex* approximation of $\gamma(z)$.

Proposition 4: (1) $0.929 \leq \gamma(z)/\gamma_1(z) \leq 1.26$;
(2) $\gamma_1(z)$ is strictly convex.

This proposition is easily proved using Proposition 3 [13]. Fig. 5 illustrates that $\gamma_1(z)$ is a good approximation of $\gamma(z)$. Now, we can obtain an approximate solution of (B). Consider the following problem (B1):

(B1) Min $\Psi(u)$
such that $\prod_{i=0}^{d(n)-1} z(H^{(i)}(n)) \geq \Lambda, \forall n \in LF$,

$$\eta_1(n, \overrightarrow{z}) = A(n) + \sum_{i \in M(n)} P(n,i)\gamma_1(z(i)) + \sum_{i \in D(n)} Q(n,i) \prod_{k=0}^{d(i)-d(n)-1} z(H^{(k)}(i)) \leq \xi(n)u, \forall n \in S.$$

The only difference between (B) and (B1) is that in (B1), $\gamma(\cdot)$ is replaced by $\gamma_1(\cdot)$. The following proposition shows that the solution of (B1) is an approximate solution of (B).

Proposition 5: Let $(\overrightarrow{z^*}, u^*)$ be the optimal solution to (B), $(\overrightarrow{z_1^*}, u_1^*)$ be the optimal solution to (B1), $T_L(\overrightarrow{z^*})$ be the network lifetime when using $\overrightarrow{z^*}$ as the capture probability thresholds, $T_L(\overrightarrow{z_1^*})$ be the network lifetime when using $\overrightarrow{z_1^*}$ as the capture probability thresholds, then $T_L(\overrightarrow{z_1^*}) \geq 0.73 T_L(\overrightarrow{z^*})$.

Proof: From Proposition 4, $0.929 \leq \frac{\gamma(z)}{\gamma_1(z)} \leq 1.26$. Therefore,

$$0.929 \leq \eta(n, \overrightarrow{z})/\eta_1(n, \overrightarrow{z})) \leq 1.26. \quad (13)$$

Because $(\overrightarrow{z_1^*}, u_1^*)$ is the optimal solution of (B1), we have

$$\eta_1(n, \overrightarrow{z_1^*}) \leq \xi(n)u_1^*, \forall n \in S.$$

Therefore, $\eta(n, \overrightarrow{z_1^*}) \leq 1.26\eta_1(n, \overrightarrow{z_1^*}) \leq 1.26\xi(n)u_1^*, \forall n \in S$. Hence,

$$T_L(\overrightarrow{z_1^*}) \geq 1/(1.26u_1^*). \quad (14)$$

Also, as $(\overrightarrow{z_1^*}, u_1^*)$ is the *optimal* solution of (B1), there must exist some node i such that $\eta_1(i, \overrightarrow{z^*}) \geq \xi(i)u_1^*$. Otherwise if $\eta_1(n, \overrightarrow{z^*}) < \xi(n)u_1^*, \forall n \in S$, then let $u_1^{'} = \max_{n \in S}\{\eta_1(n, \overrightarrow{z^*})/\xi(n)\}$. It can be easily verified that $(\overrightarrow{z^*}, u_1^{'})$ is a solution to (B1) and $u_1^{'} < u_1^*$, which is contradictory to the fact that $(\overrightarrow{z_1^*}, u_1^*)$ is the *optimal* solution of (B1).

For this node i, we have $\eta(i, \overrightarrow{z^*}) \geq 0.929\eta_1(i, \overrightarrow{z^*}) \geq 0.929\xi(i)u_1^*$, thus $T_L(\overrightarrow{z^*}) \leq 1/(0.929u_1^*)$. Combined with Eq. (14), we have $T_L(\overrightarrow{z_1^*}) \geq 0.73 T_L(\overrightarrow{z^*})$. ■

The intuition behind the proof is that $\gamma_1(\cdot)$ approximating $\gamma(\cdot)$ implies $\eta_1(n, \overrightarrow{z}) \approx \eta(n, \overrightarrow{z}), \forall n \in S$. Hence, $T_L(\overrightarrow{z^*}) = \min_{n \in S}\{\xi(n)/\eta(n, \overrightarrow{z^*})\} \approx \min_{n \in S}\{\xi(n)/\eta_1(n, \overrightarrow{z^*})\}$, and $T_L(\overrightarrow{z_1^*}) = \min_{n \in S}\{\xi(n)/\eta(n, \overrightarrow{z_1^*})\} \approx \min_{n \in S}\{\xi(n)/\eta_1(n, \overrightarrow{z_1^*})\}$. But $\overrightarrow{z_1^*}$ is the optimal solution of (B1), so $\min_{n \in S}\{\xi(n)/\eta_1(n, \overrightarrow{z_1^*})\} \geq \min_{n \in S}\{\xi(n)/\eta_1(n, \overrightarrow{z^*})\}$. Therefore, $T_L(\overrightarrow{z_1^*}) \approx \min_{n \in S}\{\xi(n)/\eta_1(n, \overrightarrow{z_1^*})\}$ cannot be much smaller than $T_L(\overrightarrow{z^*}) \approx \min_{n \in S}\{\xi(n)/\eta_1(n, \overrightarrow{z^*})\}$.

Proposition 5 is important as it shows that $\overrightarrow{z_1^*}$ is an approximate solution of (B) with *approximation ratio* 0.73.

As described earlier, (B) is a non-convex optimization problem; hence it is difficult to obtain the optimal solution $\overrightarrow{z^*}$. However, Proposition 5 shows that if we can solve (B1) and use its solution $\overrightarrow{z_1^*}$ as the capture probability thresholds, then the achieved network lifetime is no less than 73% of the maximum. Next we solve (B1).

Using the variable transformation: $v(i) = \ln(z(i))$, problem (B1) becomes the following equivalent problem (B1'):

(B1') Min $\Psi(u)$
such that $\sum_{i=0}^{d(n)-1} v(H^{(i)}(n)) \geq \ln \Lambda, \forall n \in LF$,

$$\eta_1'(n, \overrightarrow{v}) = A(n) + \sum_{i \in M(n)} P(n,i)\gamma_1(e^{v(i)}) + \sum_{i \in D(n)} Q(n,i) e^{\sum_{k=0}^{d(i)-d(n)-1} v(H^{(k)}(i))} \leq \xi(n)u, \forall n \in S.$$

In (B1'), obviously the optimization goal function is convex and the first set of constraints corresponds to a convex set. For the second set of constraints, because both $\exp(\cdot)$ and $\gamma_1(\cdot)$ are strictly convex and increasing, from the composition rule [23], $\gamma_1(\exp(\cdot))$ is also strictly convex. Therefore, the second set of constraints also corresponds to a convex set, and (B1') is a convex equivalent of (B1).

We solve (B1') via dual formulation. The dual problem is

$$\max_{\overrightarrow{\lambda} \geq 0, \overrightarrow{\mu} \geq 0} \Phi(\overrightarrow{\lambda}, \overrightarrow{\mu}),$$

where $\overrightarrow{\lambda}, \overrightarrow{\mu}$ are Lagrange multipliers corresponding to the two sets of constraints in (B1'), and $\Phi(\overrightarrow{\lambda}, \overrightarrow{\mu})$ is the dual function given by

$$\Phi(\overrightarrow{\lambda}, \overrightarrow{\mu}) = \min_{u \geq 0, \overrightarrow{v} < 0} \Psi(u) + \sum_{n \in LF} \lambda_n(\ln \Lambda - \sum_{i=0}^{d(n)-1} v(H^{(i)}(n))) + \sum_{n \in S} \mu_n(\eta_1'(n, \overrightarrow{v}) - \xi(n)u). \quad (15)$$

We use the subgradient method [23] to solve the dual problem. Let $u^*, \overrightarrow{v^*}$ be the minimizer in Eq. (15). One subgradient of the negative dual function $-\Phi(\overrightarrow{\lambda}, \overrightarrow{\mu})$ is [23]

$\vartheta_n = \sum_{i=0}^{d(n)-1} v^*(H^{(i)}(n)) - \ln \Lambda, \forall n \in LF$,
$\varphi_n = \xi(n)u^* - \eta_1'(n, \overrightarrow{v^*}), \forall n \in S$,

where $\overrightarrow{\vartheta}$ and $\overrightarrow{\varphi}$ correspond to the dual variables $\overrightarrow{\lambda}$ and $\overrightarrow{\mu}$ respectively.

To obtain the optimal dual variables, the subgradient method uses the following updates at the k^{th} iteration

$$\begin{aligned} \lambda_n(k+1) &= [\lambda_n(k) - \varpi_k \vartheta_n(k)]^+, \forall n \in LF, \quad (16) \\ \mu_n(k+1) &= [\mu_n(k) - \varpi_k \varphi_n(k)]^+, \forall n \in S, \end{aligned}$$

where $[.]^+$ denotes projection on the nonnegative orthant [5], and ϖ_k is the step size. Convergence to the optimal dual variables is guaranteed if ϖ_k satisfies $\varpi_k \to 0, \sum_{k=1}^{\infty} \varpi_k = \infty$.

Here is a physical interpretation of the dual variables $\overrightarrow{\lambda}$ and $\overrightarrow{\mu}$. Consider $\overrightarrow{\lambda}$ to be the price of violating the requirement on the delivery performance, and $\overrightarrow{\mu}$ to be the price of exceeding the battery capacity. Then, $\overrightarrow{\vartheta}$ represents the safety margin before breaking the performance requirement, and $\overrightarrow{\varphi}$ represents the excess battery capacity. The updates in Eq. (16) will increase the corresponding prices if the performance requirement is violated or the average power dissipation exceeds the capacity, and reduce the prices otherwise.

C. Implementation

In many sensor systems [28], [29], the BS is a Pentium level PC, which has a high computational capability and sufficient memory compared to the sensor nodes. Further, the BS is often connected to an unlimited power supply. Hence, it is preferable for us to take advantage of the computing capabilities of the BS and let it perform the computations[6].

After the cluster hierarchy has been established, the BS informs the nodes of the systems parameters, including the epoch duration T_e, synchronization interval T_s, and message frequency T. Each node then computes $A(n), P(n,i), Q(n,i)$ and reports to the BS. The transmission is hierarchical: the cluster members compute their $A(n), P(n,i), Q(n,i)$ values, and pass them onto the CH, then the CH combines its own parameter values with those of the members and passes onto its own CH. To guarantee that these values are received by the BS, reliable data delivery mechanisms like hop-by-hop acknowledgments can be used.

The BS solves (B1) using the subgradient method and computes the capture probability thresholds, then informs the sensor nodes. The nodes then decide the wake up schedule as described in Section III-C.

We note that the computation of the optimal capture probability thresholds is done *infrequently*, i.e., the capture probability thresholds are computed only once after the cluster hierarchy is constructed. Hence, the additional message overhead is insignificant in the long run.

[5]Note that in Problem (B), because $\eta(n, \overrightarrow{z})$ increases with $\overrightarrow{z}$, it can be seen that to guarantee a larger delivery probability, higher power is needed and the lifetime will be reduced. Hence, the optimal solution(s) occurs only when the delivery probabilities equal Λ, i.e., when $\prod_{i=0}^{d(n)-1} z(H^{(i)}(n)) = \Lambda, \forall n \in LF$. Thus, when updating λ_n, the projection $[.]^+$ is unnecessary.

[6]Note that this *centralized* scheme is effective because the BS is much more powerful than the sensor nodes. If the BS has similar performance to the sensor nodes, a distributed implementation is desirable.

D. Reclustering

In our discussions thus far, the network topology is fixed at one particular cluster hierarchy. In many systems [10], [27], periodic reclustering is used to balance the load, and the network topology alternates among several cluster hierarchies. In our technical report [13], we have shown that the problem formulation can be easily extended to account for reclustering and the solution still holds after slight modifications.

E. Simulation Results

We conduct simulations to study our threshold assignment algorithm. For illustration, we consider the cluster hierarchy in Fig. 6. The initial energy for all nodes is 1 Joule. Each node will generate $l = 4bytes$ of sensing data during each transmission period. The data aggregation overhead c is 4 $bytes$; the compression ratio $r \in [0, 1]$. We set $\Lambda = 0.7$, i.e., all information should be delivered to the BS with probability ≥ 0.7. Other simulation parameters are specified in Table 2 in our technical report [13].

For the given topology, we first note that since the BS has unlimited power supply, it can always stay awake. Thus, for messages coming from node 1, the BS will always "capture" them, and we can directly set $z(1) = 1$. Further, due to symmetry, the algorithm should set $z(2) \approx z(3)$ and $z(4) \approx z(5) \approx \ldots z(11)$. Next we consider two special cases:

- $r = 1$ corresponds to the case without any compression. In this case, node 1 is the bottleneck since it has the highest relaying burden. Hence, $z(2)$ and $z(3)$ should be small such that node 1 spends less energy for receiving. Our algorithm sets $z(2) \approx z(3) \approx 0.71$ and $z(4) \approx \ldots z(11) \approx 0.99$, correctly identifying the bottleneck.
- $r = 0$ corresponds to the case where we want updates of the type min, max, and sum. Here, transmission energy is the same for all the nodes, and the receiving energy decides the lifetime for each node. Thus, nodes 2 and 3 become the bottleneck since they need to receive from more member nodes. Correspondingly, our algorithm sets $z(2) \approx z(3) \approx 0.999$ and $z(4) \approx \ldots z(11) \approx 0.703$.

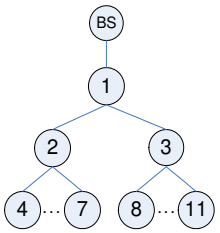

Fig. 6. Simulation topology

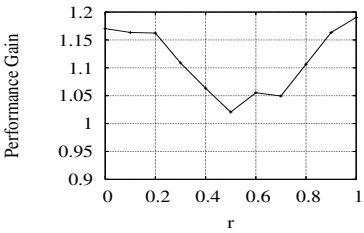

Fig. 7. Performance gain

To illustrate the performance gain of our threshold assignment algorithm, we compare with a scheme which sets equal capture probability threshold at each hop along the cluster hierarchy, $z(2) = \ldots = z(11) = \sqrt{\Lambda}$. In Fig. 7, we vary the value of r and show the performance gain, which is defined as the ratio between the network lifetime with the two schemes. We observe that our scheme always outperforms the scheme with equal thresholds. As r increases from 0 to 1,

the gain first decreases and then increases. This is because, from the above discussion, when $r = 0$ nodes 2 and 3 are the bottlenecks; hence our scheme sets $z(4), \ldots, z(11)$ to be small and $z(2), z(3)$ to be large. As r increases from 0, node 1 has a higher burden of relaying. To balance the energy consumption, our scheme increases $z(4), \ldots, z(11)$ and decreases $z(2), z(3)$. Consequently, our solution becomes closer to the scheme with equal thresholds. When $r = 0.5$, our solution almost overlaps with the other scheme and the performance gain is relatively small. But as r increases further, our solution diverges from the other scheme and achieves a higher gain, which is as large as 19% when $r = 1$. This confirms that it is necessary to adopt an intelligent scheme to assign the thresholds, and validates the effectiveness of our scheme.

V. Conclusions and Future Work

We have studied sleep/wake scheduling for low duty cycle sensor networks. Our work is different from most previous work in that we explicitly consider the effect of synchronization error in the design of sleep/wake scheduling algorithm. In our previous work [3], we showed that the impact of synchronization error is non-negligible, and studied sleep/wake scheduling for *single hop* communications. We proposed an optimal sleep/wake scheduling algorithm, which achieves a *given* capture probability threshold with minimum energy consumption.

In this work, we considered multi-hop communications. We relaxed the assumption that the capture probability threshold is *already given*, and studied how to determine the per-hop capture probability thresholds to meet the QoS requirement of the application. We observe that in many sensor networks for continuous monitoring applications, the QoS is decided by the amount of data delivered from the nodes to the base station(s), i.e., the multi-hop delivery performance. We formulate an optimization problem that sets the capture probability threshold at each hop such that the network lifetime is maximized, and yet the multi-hop delivery performance is guaranteed. The problem turns out to be non-convex and hard to solve exactly. However, by investigating its unique structure, we have obtained a 0.73-approximation algorithm. Simulations show that our solution correctly identifies the bottleneck and significantly extends the network lifetime.

In this work, we have fixed the synchronization scheme and only focused on energy conservation with sleep/wake scheduling. Synchronization and scheduling are, however, closely tied to each other and will both affect the overall system performance. Therefore, it is necessary to jointly consider synchronization and scheduling to improve the overall system performance. Further, the definition of network lifetime in this work mainly applies to application scenarios with strict coverage requirements. We plan to extend our framework to consider other definitions of network lifetime, e.g., time until network partitioning.

References

[1] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring," in *Proc. of ACM WSNA*, September 2002.

[2] N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A Wireless Sensor Network for Structural Monitoring," in *Proc. of ACM SenSys*, 2004.

[3] Y. Wu, S. Fahmy, and N. B. Shroff, "Optimal sleep/wake scheduling for time-synchronized sensor networks with qos guarantees," in *Proc. of IEEE IWQoS*, June 2006.

[4] S. Singh and C. Raghavendra, "PAMAS: Power Aware Multi-Access protocol with Signalling for Ad Hoc Networks," *ACM Computer Communication Review*, vol. 28, no. 3, pp. 5–26, July 1998.

[5] W. Ye, J. Heidenmann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," in *Proc. of IEEE INFOCOM*, New York, NY, June 2002.

[6] T. Dam and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks," in *Proc. of ACM SenSys*, 2003.

[7] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, and K. S. J. Pister, "System Architecture Directions for Networked Sensors," in *Proc. of ACM ASPLOS*, 2000, pp. 93–104.

[8] http://www.tinyos.net/scoop/special/hardware.

[9] B. Hohlt, L. Doherty, and E. Brewer, "Flexible power scheduling for sensor networks," in *Proc. of IEEE/ACM IPSN*, 2004.

[10] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, October 2002.

[11] S. Bandyopadhyay and E. Coyle, "An Energy-Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks," in *Proc. of IEEE INFOCOM*, April 2003.

[12] V. Mhatre and C. Rosenberg, "Design Guidelines for Wireless Sensor Networks Communication: Clustering and Aggregation," *Ad-hoc Networks Journal*, vol. 2, no. 1, pp. 45–63, 2004.

[13] Y. Wu, S. Fahmy, and N. B. Shroff, "Energy Efficient Sleep/Wake Scheduling for Time-Synchronized Multi-hop Sensor Networks," Technical Report, 2006, available at http://www.cs.purdue.edu/homes/wu26/techrep-multi.pdf.

[14] S. Ganeriwal, R. Kumar, and M. Srivastava, "Timing-sync Protocol for Sensor Networks," in *Proc. of ACM SenSys*, 2003.

[15] "CC1000 low power FSK transciever," Chipcon Corporation. http://www.chipcon.com.

[16] J. Elson, L. Girod, and D. Estrin, "Fine-Grained Network Time synchronization Using Reference Broadcasts," in *Proc. of USENIX/ACM OSDI*, 2002.

[17] ——, "Short Paper: A Wireless Time-Synchronized COTS Sensor Platform, Part I: System Architecture," in *Proc. of the IEEE CAS Workshop on Wireless Communications and Networking*, September 2002.

[18] Q. Gao, K. J. Blow, and D. J. Holding, "Simple algorithm for improving time synchronization in wireless sensor networks," *Electronics Letters*, vol. 40, pp. 889–891, July 2004.

[19] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The flooding time synchronization protocol," in *Proc. of ACM SenSys*, 2004.

[20] S. Ganeriwal, D. Ganesan, H. Shim, V. Tsiatsis, and M. B. Srivastava, "Estimating Clock Uncertainty for Efficient Duty-Cycling in Sensor Networks," in *Proc. of ACM SenSys*, 2005.

[21] J. R. Vig, "Introduction to Quartz Frequency Standards," Technical Report SLCET-TR-92-1, Army Research Laboratory, October 1992, available at http://www.ieee-uffc.org/freqcontrol/quartz/vig/vigtoc.htm.

[22] J. L. Devore, *Probability and Statistics for Engineering and the Sciences*. International Thomson Publishing Inc., 1995.

[23] D. P. Bertsekas, *Nonlinear programming*. Athena Scientific, 1999.

[24] E. K. P. Chong and S. H. Zak, *An Introduction to Optimization*. John Wiley & Sons, Inc., 2001.

[25] H. Nama, M. Chiang, and N. Mandayam, "Optimal utility-lifetime tradeoff in self-regulating wireless sensor networks: A distributed approach," in *Proc. of 40th Conference on Information Sciences and Systems*, 2006.

[26] R. Madan, S. Cui, S. Lall, and A. Goldsmith, "Cross-layer design for lifetime maximization in interference-limited wireless sensor networks," in *Proc. of IEEE INFOCOM*, 2005.

[27] O. Younis and S. Fahmy, "Distributed Clustering in Ad-hoc Sensor Networks: A Hybrid, Energy-Efficient Approach," in *Proc. of IEEE INFOCOM*, Hong Kong, March 2004.

[28] G. Hackmann, C.-L. Fok, G.-C. Roman, and C. Lu, "Middleware support for seamless integration of sensor and IP networks," in *Proc. of IEEE DCOSS*, 2006.

[29] L. Ho, M. Moh, Z. Walker, T. Hamada, and C. Su, "A prototype on RFID and sensor networks for elder healthcare: progress report," in *Proc. of ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis*, 2005.