Utilizing Path Diversity via Asynchronous and Asymmetric Wakeups in Sensor Networks

Anuj Rawat and Mark Shayman
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20742
Email: {anuj, shayman}@glue.umd.edu

Abstract— **We present an asynchronous wakeup policy for wireless sensor networks that exploits the available path diversity for maximizing the expected network lifetime. We assume a random traffic generation model such that the rate is constant in time. Each node is assumed to have a set of forwarding neighbors, any of which may be used for forwarding its traffic to the sink. A node having data packet to send, transmits the packet to the first available node in its forwarding set. In order to maximize the network lifetime, we balance the power dissipation at the network nodes by adjusting the wakeup parameters at various nodes. Allowing different nodes to wakeup with different rates makes the scheme asymmetric. For ease of analysis, we restrict ourselves to static, open-loop policies. We show that the optimization problem is a Signomial Program (SP), that can be well approximated as a Geometric Program (GP). By extensive simulations, we compare the asymmetric policy thus obtained to the best possible symmetric policy obtained from the same optimization setup but ensuring additionally that the wakeup rates at all the nodes are the same (in which case the optimization problem is shown to be exactly a GP). The simulations show that allowing asymmetry can extend the network lifetime by effectively exploiting the available path diversity. Moreover, we also prove that, in case of symmetric policies, no piecewise static policy can beat the simple static policy that we use for comparison in our results. This shows that in the space of open-loop, asynchronous wakeup policies, employing the static, asymmetric policy presented in this paper is much more profitable than even the best piecewise static, symmetric policy.**

I. Introduction

Maximizing the lifetime of wireless sensor networks has been an area of active research for some time. Two orthogonal strategies for the problem are either to conserve the battery at individual nodes, or to employ energy harvesting schemes. We study the problem of maximizing the network lifetime by conserving the nodes' battery. For this problem, several approaches, such as energy aware routing, in-network data aggregation, duty cycling, adaptive sensing, etc., have been proposed. In this paper, we focus on the duty cycling approach. The intuition for employing duty cycling is that if a node is idle, then its radio may be turned off, since idle listening causes substantial energy drain.

Various duty cycling schemes (also referred to as wakeup or sleep schemes) have been studied in the past. These schemes can be broadly classified as synchronous, asynchronous, and on-demand wakeups. As the name suggests, in synchronous wakeup schemes, all the network nodes wake up at the same, predetermined time. This approach is attractive because it is possible to support extremely short duty cycles. The drawback is that achieving and maintaining clock synchronization in distributed systems is not a trivial task, and also, it is hard to deploy dynamic synchronous wakeup policies that can adapt to random perturbations/changes in the system. Examples of wakeup schemes based on synchronization include S-MAC [1], [2] and T-MAC [3]. In on-demand wakeup schemes, nodes are equipped with an additional, low-power radio, that is never powered off. Using this low-power radio, a transmitter can request the intended receiver to power on its primary radio. Although this scheme does away with the synchronization problem, it has its own drawbacks, such as because of the multiple radios, the nodes are more expensive, a part of the available bandwidth is dedicated for operating the low-power control radios, and usually the transmission range of the primary and the low-power radios are not the same. Examples of on-demand wakeup schemes include STEM [4], rate estimated MAC [5], and passive-radio triggered wakeup schemes [6]. Asynchronous wakeup schemes remove the need of synchronization by ensuring that the neighboring nodes are able to detect each other in finite time. In asynchronous wakeup schemes such as AWP [7], the schemes presented in [8], etc., this is ensured by selecting the wakeup and sleep schedules at various nodes so that any two neighboring nodes are guaranteed to have overlapping active periods in finite time. Another approach, followed in schemes such as B-MAC [9], X-MAC [10], SEESAW [11], [12] etc., requires that the sleeping nodes periodically check the channel for any activity, and the sender initiates communication with the intended receiver by transmitting a (strobed train of) header or request packet(s) which may be detected at the receiver during some periodic channel listen. Since we are interested in asynchronous wakeup schemes employing header packets and periodic channel listens, we only discuss these schemes in more detail. The interested reader is referred to [13], [14] for a survey of wakeup schemes employed in sensor networks.

Low Power Listen (LPL) operation, described in [9], allows nodes to check any activity on the channel by sampling it for a small time. This is an extremely low energy operation and forms the main idea behind schemes such as B-MAC, X-MAC, etc. In B-MAC, nodes periodically perform LPL checks with a fixed period that is the same for all the nodes. This is referred to as the LPL check interval. In order to guarantee packet delivery, the sender transmits the data packet

Research partially supported by the NSF under grant CNS-0519554

with a preamble that is longer than the LPL check interval. X-MAC improves upon B-MAC by allowing the intended receiver to acknowledge its readiness. This allows the sender to start transmitting the packet without sending the full long preamble, thereby improving per-hop latency as well as the energy efficiency. In SEESAW, idle nodes periodically listen to the channel. The sender transmit a train of uniformly spaced advertisements to initiate communication with the intended receiver. The fraction of time an idle node spends listening to the channel, and the spacing between advertisements transmitted by a sender are node parameters and may be different for different nodes. In this sense, SEESAW is an asymmetric wakeup scheme as opposed to symmetric wakeup schemes like B-MAC. SEESAW tries to balance the energy spent in protocol overheads at various nodes by exploiting this asymmetry, i.e., by adjusting the two parameters at various nodes, in order to maximize the network lifetime. SEESAW assumes a single available route to the sink from every node, hence the data rate seen by individual nodes is solely determined by the data generation process and is independent of the wakeup scheme.

In this paper, we assume that the network nodes may have multiple available paths for reaching the sink. This is true in most of the sensor networks (expect for very sparse networks). An upshot of this is that the path diversity available may be used to balance not only the protocol overheads, but also the actual data traffic as seen by various network nodes. For this end, we propose the use of a simple asymmetric, asynchronous wakeup scheme. The basic idea is that it is possible to configure the LPL check rates of the nodes such that in a set of possible receivers for a sender, the receivers that do not see much traffic (from their other senders) wake up faster, so that they share a larger portion of the traffic from this sender, thus alleviating the pressure on the receivers that are seeing large amounts of traffic from their other senders. RAW [15], a symmetric, asynchronous wakeup protocol, achieves improved latency and network lifetime by utilizing the available path diversity. The problem with RAW is that there may be packet losses. Moreover, [15] provides no analysis of the network lifetime achieved by RAW and neither does it provide any insight or discussion on how to select the policy parameters for achieving maximum lifetime improvement. Also, as we shall see later, the symmetric assumption restricts the protocol from utilizing the true potential of path diversity in the network.

In this paper, we assume a network where the data traffic is generated according to a homogeneous Poisson process. We analyze the performance of static, open-loop, asynchronous wakeup schemes. We construct the problem of determining the wakeup rates in order to maximize the network lifetime as a Signomial Program (SP) [16] which, in general, is not a convex optimization problem. But in our case, the problem turns out to be very close to a Geometric Program (GP) [16], and can therefore be solved approximately. We compare this wakeup policy with the best possible symmetric policy, which is shown to be the solution of a GP, and therefore easily solvable. The simulations show that the asymmetry does indeed buy us a lot of leverage for extending the network lifetime. Moreover we also prove that, in case of symmetric policies, no piecewise static policy can beat the simple static policy obtained by solving the constructed GP. This shows that in the space of open-loop, asynchronous wakeup policies, employing static, asymmetric policy is much more profitable than even the best piecewise static, symmetric policy (which may itself be hard to find, since the number of pieces is also an unknown).

II. Communication Model

A. Network and Data Generation

We denote the set of network nodes by $\mathbf{V} = \{0, 1, \ldots, N\}$, where node 0 is the *sink* and the rest of the nodes (referred to as *sensors*) can act as both data sources and relays. The set of sensors is denoted by $\mathbf{S}$. The adjacency information of the network nodes is modeled as the undirected graph $\mathbf{G}$ with vertex set $\mathbf{V}$ and the edge set $\mathbf{E_G}$ representing the pairs of adjacent nodes. An edge between nodes $u, v \in \mathbf{V}$, is denoted by $\{u, v\} \in \mathbf{E_G}$. We assume that any two nodes can directly communicate with each other if and only if they are adjacent. We denote the set of all nodes adjacent to node $v \in \mathbf{V}$ by $\mathbf{N}_v$, i.e., $\mathbf{N}_v = \{u \in \mathbf{V} : \{u, v\} \in \mathbf{E_G}\}$. This is also referred to as the set of *neighbors* of network node v. We assume that if multiple neighbors of a node $v \in \mathbf{V}$ transmit simultaneously, then if v listens to the channel, it shall hear a collision.

We assume that no data packets are generated at the sink, and at every sensor, data packets are generated according to a homogeneous Poisson process. We assume that the packet generation processes for distinct sensors are independent and may have distinct rates.

B. Routing

We assume that the routing is predetermined in the sense that for every sensor, the set of possible next hop neighbors is fixed. This is modeled as a directed graph $\vec{\mathbf{R}}$ with vertex set $\mathbf{V}$ and directed edge set $\mathbf{E}_{\vec{\mathbf{R}}}$ where for a pair of nodes $u, v \in \mathbf{V}$, the presence of a directed edge from node u to node v, denoted by $(u, v) \in \mathbf{E}_{\vec{\mathbf{R}}}$, implies that node v is in the set of possible next hop neighbors of node u. We assume that $\vec{\mathbf{R}}$ is a Directed Acyclic Graph (DAG) with a strictly positive outdegree for every sensor and outdegree equal to zero for the sink. This ensures that there are no routing loops and that every sensor has a directed path to the sink.

We denote the set of all the forwarding neighbors of sensor $v \in \mathbf{S}$ by $\mathbf{D}_v$, i.e., $\mathbf{D}_v = \{u \in \mathbf{V} : (v, u) \in \mathbf{E}_{\vec{\mathbf{R}}}\}$. This is also referred to as the set of *downstream neighbors* of sensor v. By $\mathbf{U}_v$, we denote the set of all the sensors for which the network node $v \in \mathbf{V}$ acts as a forwarding neighbor, i.e., $\mathbf{U}_v = \{u \in \mathbf{S} : (u, v) \in \mathbf{E}_{\vec{\mathbf{R}}}\}$. This is also referred to as the set of *upstream neighbors* of network node v.

C. Node State

Since usually there is no energy constraint at the sink, we assume that the sink always listens to the channel. On the other hand, the state of a sensor is determined by the state of its radio. At any time t, a sensor $v \in \mathbf{S}$ is in one of the following four states:

(i) SLEEP (S): In this state, v's radio is OFF.

(ii) RECEIVE (R): In this state, v is in receive mode, i.e., it listens to the channel for a data packet from one of its upstream neighbors.

(iii) TRANSMIT (T): In this state, v is in data transmit mode, i.e., it broadcasts a data packet on the channel for one of its downstream neighbors.

(iv) HEADER (H): In this state, v is in header transmit mode, i.e., it broadcasts header and also listens to the channel for any response to its header transmission, from any of its neighboring nodes.

While in state S, sensor v may check the channel for any sort of activity by performing LPL checks. The length of time a sensor stays in state S, between any two successive LPL checks, is referred to as *LPL check interval*. We assume that the LPL check intervals at sensor v are independent and exponentially distributed with parameter $\frac{1}{\lambda^{W_v}}$. We refer to λ^{W_v} as the *wakeup rate* of the sensor v. Moreover, the LPL check intervals at distinct sensors are assumed to be independent. During an LPL check, if sensor v detects a collision, it broadcasts a NAK; on the other hand if it detects that one of its upstream neighbors is transmitting header, it broadcasts an ACK and switches to state R in order to receive a data packet from the upstream neighbor. The sensor v remains in state R until it successfully receives the data packet. On successfully receiving the data packet, it broadcasts an ACK. All this is true for the sink also, i.e., if the sink detects a collision, it broadcasts a NAK, and if it detects that one of its upstream neighbors is transmitting header, it broadcasts an ACK and prepares to receive a data packet from the upstream neighbor. On successfully receiving the data packet, it broadcasts an ACK.[1]

At any time t, a sensor v tries to grab the channel in order to initiate a packet transmission with probability 1 if v was in state S at time $t-\mathrm{d}t$, and a new data packet was generated at v during the time interval $(t-\mathrm{d}t, t)$, and with probability $p\,\mathrm{d}t$ if v was in state S at time $t-\mathrm{d}t$, and there were packets in its buffer awaiting transmission. In the latter case, p is referred to as the *persistence* of the communication model. If sensor v successfully grabs the channel, it goes into state H. It remains in state H until it receives an ACK from one of its downstream neighbors, or NAKs from some of its neighboring nodes. If sensor v receives at least one NAK, it goes into state S. On the other hand if it only receives an ACK from one of its downstream neighbors (say $u \in \mathbf{D}_v$), it goes into state T and keeps on broadcasting the packet until it receives another ACK from node u confirming the reception of the data packet.

III. WAKEUP ANALYSIS

The problem that we wish to address is, given a network graph $\mathbf{G}$, routing DAG $\vec{\mathbf{R}}$ and the data generation rates at the sensors, determine the wakeup rate λ^{W_v} for every sensor $v \in \mathbf{S}$ in order to maximize the expected lifetime of the network. Here we quantify the lifetime of the network as the time till the first sensor fails. This concept of lifetime is widely used in sensor network literature, and has the justification that if any sensor dies, the sink no longer gets the complete profile of the region being observed by the sensor network.

Let $J_v(t)$ be the energy dissipated at sensor v up to time t, and let $P_v(t)$ be the power drain at sensor v at time t. For every sensor $v \in \mathbf{S}$, we have

$$J_v(t) = \int_0^t P_v(s)\,\mathrm{d}s. \tag{1}$$

Since the lifetime of the sensor is long compared to the timescales at which traffic is generated and since we employ static wakeup policy, we assume that after a small transient period, the system achieves stationarity. In particular, assuming a stationary, ergodic framework, for every sensor $v \in \mathbf{S}$, random variables $\{P_v(t)\}_{t\geq 0}$ are distributed identically to a generic random variable P_v. Hence from (1), we have

$$\lim_{t\uparrow\infty} \frac{J_v(t)}{t} = \lim_{t\uparrow\infty} \frac{1}{t}\int_0^t P_v(s)\,\mathrm{d}s = \mathbb{E}[P_v] \quad a.s. \tag{2}$$

where the notation $\mathbb{E}[R]$ denotes the expected value of random variable R. Let E_{init} be the initial energy at each sensor, and $E_v(t)$ be the residual energy at sensor v after time t. From (1), we have

$$\mathbb{E}[E_v(t)] = E_{\text{init}} - \mathbb{E}[J_v(t)] = E_{\text{init}} - t\,\mathbb{E}[P_v]. \tag{3}$$

From (3), the expected lifetime of the network, defined as the time at which the expected residual energy at any sensor vanishes, is given by

$$T = \min_{v\in\mathbf{S}} \frac{E_{\text{init}}}{\mathbb{E}[P_v]}. \tag{4}$$

Hence, the objective of maximizing the average network lifetime is equivalent to minimizing the maximum average power dissipation over the set of all the sensors.

The rate of any given counting process C is defined as

$$\lambda^C \triangleq \lim_{t\uparrow\infty} \frac{C(t)}{t}.$$

For any sensor $v \in \mathcal{S}$, let A_v be the counting process associated with the packet arrivals at v from its upstream nodes, and let G_v be the counting process associated with the packet generation at sensor v. Let the counting process X_v be the sum of A_v and G_v. Therefore, the associated rates satisfy

$$\lambda^{X_v} = \lambda^{A_v} + \lambda^{G_v}. \tag{5}$$

Let L_v be the counting process associated with the LPL operation at sensor v, with rate λ^{L_v}.

At any node, let the energy spent in transmitting a packet be e_{tx}, the energy spent in receiving a packet be e_{rx}, the energy spent in generating a packet be e_{gen}, the energy spent in performing a low-power listen operation be e_{lpl}, and the

[1] It should be stated that a sensor $v \in \mathbf{S}$ having $\mathbf{D}_v = \{0\}$, does not need to transmit header since the intended receiver (sink) is already listening. This alternate behavior has no bearing on the discussion, analysis and results presented in this paper. We do not consider this alternate behavior for ease of exposition.

power spent during header transmission be $p_{\rm hdr}$. The energy dissipated at any sensor $v \in \mathcal{S}$ up to time t can be written as

$$J_v(t) = \sum_{i=1}^{X_v(t)} Q_v^i + \sum_{j=1}^{L_v(t)} e_{\rm lpl}, \tag{6}$$

where Q_v^i is the energy spent by the i-th packet being transmitted by sensor v. Let H_v^i be the length of header transmitted by sensor v before successfully transmitting the i-th packet. Then,

$$Q_v^i = \left\{ \begin{array}{ll} e_{\rm rx} + e_{\rm tx} + p_{\rm hdr} H_v^i & \text{if } i\text{-th packet arrives at } v, \\ e_{\rm gen} + e_{\rm tx} + p_{\rm hdr} H_v^i & \text{otherwise.} \end{array} \right.$$

We assume that the sets of random variables $\{Q_v^i\}_i$ and $\{H_v^i\}_i$ are independent and distributed identically to the random variables Q_v and H_v, respectively. Hence,

$$\mathbb{E}[Q_v] = e_{\rm rx} \frac{\lambda^{A_v}}{\lambda^{X_v}} + e_{\rm gen} \frac{\lambda^{G_v}}{\lambda^{X_v}} + e_{\rm tx} + p_{\rm hdr} \mathbb{E}[H_v]. \tag{7}$$

We assume that the data packets are of equal lengths and denote the length of packet transmission by $T_{\rm pkt}$. Let t_v^I be the total time that sensor v is idle, i.e., not transmitting or receiving any packets or headers, till time t. Then

$$t_v^I = t - \sum_{i=1}^{X_v(t)} \left(H_v^i + T_{\rm pkt} \right) - \sum_{j=1}^{A_v(t)} T_{\rm pkt}. \tag{8}$$

From (8), we have

$$\begin{aligned} \lim_{t \uparrow \infty} \frac{t_v^I}{t} =& 1 - \lim_{t \uparrow \infty} \frac{X_v(t)}{t} \left(\frac{\sum_{i=1}^{X_v(t)} H_v^i}{X_v(t)} + T_{\rm pkt} \right) \\ & - T_{\rm pkt} \lim_{t \uparrow \infty} \frac{A_v(t)}{t} \\ =& 1 - \lambda^{X_v} \left(\mathbb{E}[H_v] + T_{\rm pkt} \right) - \lambda^{A_v} T_{\rm pkt}. \end{aligned} \tag{9}$$

Since during its idle time, sensor v performs the LPL operations with rate λ^{W_v}, using (9), the overall rate of the LPL operations performed by v is given as

$$\begin{aligned} \lambda^{L_v} =& \lim_{t \uparrow \infty} \frac{L_v(t)}{t} = \lim_{t \uparrow \infty} \frac{t_v^I}{t} . \frac{L_v(t)}{t_v^I} \\ =& \lambda^{W_v} \left(1 - \lambda^{X_v} \left(\mathbb{E}[H_v] + T_{\rm pkt} \right) - \lambda^{A_v} T_{\rm pkt} \right). \end{aligned} \tag{10}$$

For every sensor $v \in \mathcal{S}$, (2), (6), (7) and (10) imply

$$\begin{aligned} \mathbb{E}[P_v] =& \lim_{t \uparrow \infty} \frac{1}{t} \left(\sum_{i=1}^{X_v(t)} Q_v^i + \sum_{j=1}^{L_v(t)} e_{\rm lpl} \right) \\ =& \lim_{t \uparrow \infty} \frac{X_v(t)}{t} . \frac{\sum_{i=1}^{X_v(t)} Q_v^i}{X_v(t)} + \lim_{t \uparrow \infty} e_{\rm lpl} \frac{L_v(t)}{t} \\ =& \lambda^{X_v} \mathbb{E}[Q_v] + \lambda^{L_v} e_{\rm lpl} \\ =& e_{\rm tx} \lambda^{X_v} + e_{\rm rx} \lambda^{A_v} + e_{\rm gen} \lambda^{G_v} + p_{\rm hdr} \lambda^{X_v} \mathbb{E}[H_v] + \\ & \lambda^{W_v} e_{\rm lpl} \left(1 - \lambda^{X_v} (\mathbb{E}[H_v] + T_{\rm pkt}) - \lambda^{A_v} T_{\rm pkt} \right). \end{aligned} \tag{11}$$

Assuming that during the idle state of sensor v, the LPL process is a Poisson arrival process, we obtain

$$\mathbb{E}[H_v] = \sum_{i \in \mathbf{D}_v} \frac{1}{\lambda^{W_v}}. \tag{12}$$

Moreover, by the steady state assumption,

$$\begin{aligned} \lambda^{A_v} &= \sum_{w \in \mathbf{U}_v} \lambda^{X_w} \frac{\lambda^{W_v}}{\sum_{x \in \mathbf{D}_w} \lambda^{W_x}} \\ &= \lambda^{W_v} \sum_{w \in \mathbf{U}_v} \lambda^{X_w} \mathbb{E}[H_w]. \end{aligned} \tag{13}$$

Let $\Lambda^W = \{\lambda^{W_v} : v \in \mathbf{S}\}$. The overall optimization problem of interest becomes

$$\begin{aligned} & \min_{\Lambda^W} \max_{v \in \mathbf{s}} \mathbb{E}[P_v] \\ \text{subject to} \quad & \text{equations (11), (12),} \\ & \text{(13) and (5), } \forall v \in \mathbf{S}. \end{aligned} \tag{14}$$

This optimization problem can be rewritten as a Signomial Program (SP). As stated in Section I, in general SP is not easy to solve. But the SP obtained from (14), is very close to a Geometric Program (GP), and can therefore be solved approximately. We use this approximate solution as our static, open-loop, asymmetric, asynchronous wakeup policy. The exact details of rewriting (14) into a SP and then approximating it with a GP are given in [17].

In symmetric wakeup schemes, for every sensor $v \in \mathcal{S}$, we have $\lambda_v^W = \lambda^W$. Hence, if we restrict the wakeup scheme to being a symmetric scheme, the optimization problem reduces to

$$\min_{\lambda^W} \max_{v \in \mathbf{s}} \mathbb{E}[P_v] \tag{15}$$

subject to

$$\begin{aligned} \mathbb{E}[P_v] =& (e_{\rm gen} - e_{\rm rx}) \lambda^{G_v} + (e_{\rm tx} + e_{\rm rx} - \frac{e_{\rm lpl}}{|\mathbf{D}_v|}) \lambda^{X_v} \\ & + e_{\rm lpl} (1 - (2\lambda^{X_v} - \lambda^{G_v}) T_{\rm pkt}) \lambda^W \\ & + \frac{p_{\rm hdr} \lambda^{X_v}}{|\mathbf{D}_v|} \frac{1}{\lambda^W}, \\ \lambda^{X_v} =& \lambda^{G_v} + \textstyle\sum_{w \in \mathbf{U}_v} \frac{\lambda^{X_w}}{|\mathbf{D}_w|}, \qquad \forall v \in \mathbf{S}. \end{aligned}$$

This optimization problem can be rewritten as a GP, which is easy to solve. Again, the details of rewriting (15) into a GP are given in [17].

Next we consider the set of all the piecewise static, open-loop, symmetric, asynchronous wakeup policies. Clearly they contain the set of static, open-loop, symmetric, asynchronous wakeup policies that we considered in (15). Let the number of pieces be M. Let the wakeup rate in piece m be λ_m^W. Let the fraction of the total lifetime that λ_m^W is used as the wakeup rate be α_m. Let $P_{v,m}$ be the random variable analogous to random variable P_v in piece m. Hence, the optimization is

$$\min_{\{\alpha_m, \lambda_m^W\}_{m=1}^M} \max_{v \in \mathbf{s}} \alpha_m \mathbb{E}[P_{v,m}] \tag{16}$$

subject to

$$\begin{aligned} \mathbb{E}[P_{v,m}] =& (e_{\rm gen} - e_{\rm rx}) \lambda^{G_v} + (e_{\rm tx} + e_{\rm rx} - \frac{e_{\rm lpl}}{|\mathbf{D}_v|}) \lambda^{X_v} \\ & + e_{\rm lpl} (1 - (2\lambda^{X_v} - \lambda^{G_v}) T_{\rm pkt}) \lambda_m^W \\ & + \frac{p_{\rm hdr} \lambda^{X_v}}{|\mathbf{D}_v|} \frac{1}{\lambda_m^W}, \\ \lambda^{X_v} =& \lambda^{G_v} + \textstyle\sum_{w \in \mathbf{U}_v} \frac{\lambda^{X_w}}{|\mathbf{D}_w|}, \qquad \forall v \in \mathbf{S}, \\ \textstyle\sum_{m=1}^M & \alpha_m = 1. \end{aligned}$$

The detailed discussion on constructing this optimization problem is given in [17]. In [17], we also show that for any fixed M, the optimization problem is a GP.

Theorem 3.1: The maximum expected lifetime achieved by the piecewise static policy obtained on solving (16) cannot beat the maximum expected lifetime achieved by the static policy obtained on solving (15).

The proof is based on the inequality relating arithmetic mean, geometric mean and harmonic mean. The complete proof is given in [17].

IV. Simulation Results

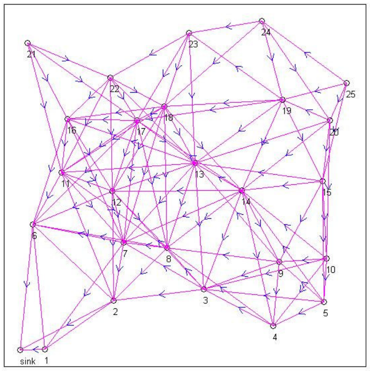

Fig. 1. Network graph $\mathbf{G}$ and routing DAG $\vec{\mathbf{R}}$

For simulations, we study a network of 25 sensors and a sink. The network graph $\mathbf{G}$ and the routing DAG $\vec{\mathbf{R}}$ were generated as described next. Consider a unit square in $\mathbb{R}_+^2$ with the origin as the bottom left corner point. Divide this square into 25 equal sized squares and label them from 1 to 25, traversing the rows of squares from the bottom to the top and traversing the squares in each row from left to right. As stated before, the set of nodes is denoted by $\mathbf{V} = \{0, 1, \ldots, 25\}$, where 0 is the sink and the rest of the nodes are sensors. The sink is placed at the origin, and the sensor i is placed uniformly randomly in the i-th small square. The network graph $\mathbf{G}$ has vertex set $\mathbf{V}$ and edge set $\mathbf{E}_\mathbf{G} = \{\{u, v\} : u, v \in \mathbf{V}, \|u - v\|_2 \leq \frac{1}{\sqrt{5}}\}$, where $\| \cdot \|_2$ is the Euclidean norm. The norm condition assures that the network graph is connected. The routing DAG is generated by using simple geographic routing where a sensor adjacent to the sink is allowed to transmit data packets only to the sink, and a sensor far away from the sink, is allowed to transmit data packets to all of its neighboring sensors that are strictly closer to the sink than itself. In other words, for any sensor $u \in \mathbf{S}$, the set of downstream neighbors is given as

$$\mathbf{D}_u = \begin{cases} \{v \in \mathbf{N}_u : \|v\| < \|u\|\} & \text{if } 0 \notin \mathbf{N}_u, \\ \{0\} & \text{otherwise.} \end{cases}$$

The actual network and the routing DAG used for the simulation (that were generated as described above) are shown in Figure 1. The solid lines represent adjacency, and the arrows on the edges represent the direction of possible data flow in case the directed edge is present in the routing DAG.

For the simulations, we assumed slotted time. We assume that a new packet is generated at any sensor $u \in \mathbf{S}$ in any timeslot k with probability 0.0005. We assume that the length of a timeslot to be 2.5 ms. Hence, the packet generation rate λ^G at the sensors is equal to 0.0005 per timeslot or 0.2 s^{-1}. Also, an LPL check requires exactly one timeslot [9]. At a data transfer rate of 100 kbps and a packet size of 250 bits, a data packet transmission/reception requires 2.5 ms, i.e., one timeslot. The total energy spent by a sensor in performing an LPL check operation is equal to 17.3 μJ. Assuming Chipcon CC1000 radio [18], the current drawn while transmitting and receiving data is equal to 25.4 mA and 9.6 mA, respectively. With a power supply of 3.0 V and the data packet length of 2.5 ms, the energy spent by a sensor in transmitting (receiving) a data packet is equal to 190.5 μJ (72 μJ). While transmitting header, the node also listens for ACKs from its downstream neighbors, and NAKs from all of its neighboring nodes. Hence, the energy spent by a sensor in transmitting header for one timeslot is equal to 262.5 μJ. If a sensor wants to transmits but cannot grab the channel, the node still has to expend 17.3 μJ amount of energy (equal to the energy required for an LPL check). In this case the node is assumed to be in idle state during that timeslot. Moreover, we assume that the energy required to generate a new packet is 500 μJ. This usually depends on the sensing application, but in most of the sensing applications, the sensing operation (data generation) itself accounts for a very small fraction of the energy depletion at the nodes. We normalize the various energy values with respect to the energy required for an LPL check. We assume that the normalized initial energy at every node is 500000 units.[2] The various system parameters used in the simulations, are presented in Table I.

TABLE I
Parameters

Parameter	Value
$\lvert\mathbf{S}\rvert$	25
E_{init}	500000 units
e_{gen}	30 units
e_{lpl}	1 unit
e_{rx}	4 units
e_{tx}	11 units
p_{hdr}	15 units per timeslot
e_{id}	1 unit
λ^G	0.0005 packets per timeslot

We simulated the performance for both the symmetric and asymmetric policies as described in Section III. In the symmetric case, the wakeup rate λ^W (interpreted in the slotted model as the probability of wakeup in any particular timeslot) was determined to be equal to 0.1568. The wakeup rates for the sensors as determined by the asymmetric policy are presented in Table II.

The simulation results are compiled in Table III and Figure

[2] Although the initial energy at nodes is much higher than this, during the simulations we observe that this value is high enough to ensure that the system achieves the stationary state.

TABLE II

WAKEUP RATES FOR ASYMMETRIC POLICY

Node Id	21	22	23	24	25
Wakeup rate	0	0.0498	0.0478	0.0412	0
Node Id	16	17	18	19	20
Wakeup rate	0.0629	0.0536	0.0435	0.0407	0.0332
Node Id	11	12	13	14	15
Wakeup rate	0.0464	0.0490	0.0416	0.0332	0.0287
Node Id	6	7	8	9	10
Wakeup rate	0.0840	0.0386	0.0247	0.0255	0.0237
Node Id	1	2	3	4	5
Wakeup rate	0.1742	0.0863	0.0331	0.0113	0.0243

2. Table III presents the average useful lifetime of the network, measured as the number of packets reaching the sink before the first node failure, over 30 random runs while employing the two wakeup policies. Figure 2 presents the residual energy profile of the network nodes under the two policies averaged over 30 random runs.

TABLE III

AVERAGE USEFUL LIFETIME

Symmetric Policy (# packets)	Asymmetric Policy (# packets)
14489	21587

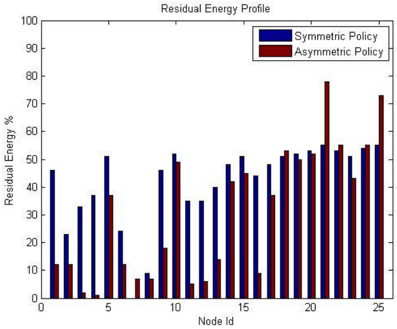

Fig. 2. Residual Energy Profile

From Table III, we observe that the asymmetric policy performs significantly better than the symmetric policy in terms of the total lifetime of the network. In fact, it increases the average network lifetime by nearly 50%. Figure 2 shows that the asymmetric policy does a much better job of balancing the energy consumption at the network nodes as compared to the symmetric policy. To observe this, note that the number of nodes with the residual energy being less than 20% of the initial energy is 12 (out of 25) in the case of asymmetric policy, as compared to only 2 in the case of symmetric policy.

V. CONCLUSION AND FUTURE WORK

In this paper, we presented a static, open-loop, asynchronous wakeup policy for wireless sensor networks, that exploits the available path diversity for maximizing the expected network lifetime. The policy is able to balance the power dissipation at various network nodes by adjusting a single parameter for that node. By simulations, we compared this asymmetric policy to the best possible symmetric policy and established that asymmetry does indeed allow us to effectively exploit the available path diversity and increase the network lifetime. We also prove that, in case of symmetric policies, no piecewise static policy can beat the simple static policy that was used for comparison in our results. This shows that in the space of open-loop, asynchronous wakeup policies, employing the static, asymmetric policy presented in this paper is much more profitable than even the best piecewise static, symmetric policy.

Next, we would like to develop and analyze a closed-loop asynchronous wakeup policy. This may be significantly harder than the problem studied in this paper, mainly because we may no longer be able to rely on the steady state analysis that was presented here. Another interesting problem that we are working on is to develop a distributed algorithm by which the various nodes can decide their wakeup rates.

REFERENCES

[1] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *IEEE Infocom*, June 2002.

[2] ——, "Medium access control with coordinated, adaptive sleeping for wireless sensor networks," *ACM/IEEE Transactions on Networking*, vol. 12, no. 3, pp. 493–506, June 2004.

[3] T. van Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," in *ACM SenSys*, 2003.

[4] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava, "Optimizing sensor networks in the energy-latency-density design space," *IEEE Transactions on Mobile Computing*, vol. 1, no. 1, pp. 70–80, January 2002.

[5] M. Miller and N. Vaidya, "A MAC protocol to reduce sensor network energy consumption using a wakeup radio," *IEEE Transactions on Mobile Computing*, vol. 4, no. 3, pp. 228–242, May 2005.

[6] L. Gu and J. A. Stankovic, "Radio-triggered wake-up capability for sensor networks," in *IEEE RTAS*, 2004.

[7] R. Zheng, J. C. Hou, and L. Sha, "Asynchronous wakeup for ad hoc networks," in *ACM MobiHoc*, 2003.

[8] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh, "Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks," in *IEEE Infocom*, 2002.

[9] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *ACM SenSys*, 2004.

[10] M. Buettner, G. Yee, E. Anderson, and R. Han, "X-MAC: A short preamble mac protocol for duty-cycled wireless sensor networks," in *ACM SenSys*, 2006.

[11] R. Braynard, S. Ravi, and C. Ellis, "Exploring the design of an asynchronous and asymmetric MAC protocol," in *SenMetrics*, 2005.

[12] R. Braynard, A. Silberstein, and C. Ellis, "Extending network lifetime using an automatically tuned energy-aware MAC protocol," in *EWSN*, 2006.

[13] T. Armstrong, "Wake-up based power management in multi-hop wireless networks," available at http://www.eecg.toronto.edu/~trevor/Wakeup/survey.pdf.

[14] K. Langendoen, "The mac alphabet soup served in wireless sensor networks," available at https://apstwo.st.ewi.tudelft.nl/~koen/MACsoup.

[15] V. Paruchuri, S. Basavaraju, A. Durresi, R. Kannan, and S. S. Iyengar, "Random asynchronous wakeup protocol for sensor networks," in *Broadnets*, 2004.

[16] M. Chiang, "Geometric programming for communication systems," *Foundations and Trends in Communications and Information Theory*, vol. 2, no. 1/2, pp. 1–154, 2005.

[17] A. Rawat and M. Shayman, "Utilizing path diversity via asynchronous and asymmetric wakeups in sensor networks," available at http://www.ece.umd.edu/~anuj/GLOBECOM/tr.pdf.

[18] "Chipcon cc1000 datasheet," available at http://focus.ti.com/lit/ds/symlink/cc1000.pdf.