

# Adaptive Low Power Listening for Wireless Sensor Networks

Raja Jurdak, *Member, IEEE*, Pierre Baldi, *Senior Member, IEEE*, and Cristina Videira Lopes, *Member, IEEE*

**Abstract**—Most sensor networks require application-specific network-wide performance guarantees, suggesting the need for global and flexible network optimization. The dynamic and nonuniform local states of individual nodes in sensor networks complicate global optimization. Here, we present a cross-layer framework for optimizing global power consumption and balancing the load in sensor networks through greedy local decisions. Our framework enables each node to use its local and neighborhood state information to adapt its routing and MAC layer behavior. The framework employs a flexible cost function at the routing layer and adaptive duty cycles at the MAC layer in order to adapt a node's behavior to its local state. We identify three state aspects that impact energy consumption: 1) number of descendants in the routing tree, 2) radio duty cycle, and 3) role. We conduct experiments on a test-bed of 14 mica2 sensor nodes to compare the state representations and to evaluate the framework's energy benefits. The experiments show that the degree of load balancing increases for expanded state representations. The experiments also reveal that all state representations in our framework reduce global power consumption in the range of one-third for a time-driven monitoring network and in the range of one-fifth for an event-driven target tracking network.

**Index Terms**—Cross-layer, framework, cost function, routing, medium access control, state information, test-bed, energy, power.

## 1 INTRODUCTION

WIRELESS sensor networks consist of autonomous sensor nodes that monitor physical indicators in their environment and communicate with each other to report the data, interfacing the physical world with the digital domain. Sensor networks present both new application opportunities and new design challenges. Sensor network applications include tracking and intrusion detection for military purposes, pollutant and habitat monitoring for environmental purposes, traffic and location monitoring for civilian use, and warehousing and toxic leak monitoring for industrial automation applications.

Each application typically requires custom performance guarantees, so the wide application space is a design challenge in and of itself. Other challenges for sensor network design include the limited resources at each node and the lack of infrastructure of these networks.

### 1.1 Design Challenges

Strictly layered communication models, such as the OSI reference model, are designed for conventional wired and wireless networks, where individual nodes have relatively large communication bandwidth, processing power, memory capacity, and energy resources. Unlike most conventional networks, the limited resources at individual nodes in sensor networks require more versatile cross-layer models

that enable fine-grained optimization of resource usage. Several cross-layer models have been proposed to address specific optimization variables in wireless sensor networks, such as link scheduling and routing flow [2], [3], [4], [5]. Here, we adopt the cross-layer optimization framework for sensor networks which we had proposed in [6], and we tailor this framework for a data gathering sensor network. Before presenting the framework, the following discussion explores the challenges that motivate the framework's design choices.

Traditionally, communication networks have not been designed from scratch in a principled way to optimize global network properties. As an emerging class of networks, sensor networks can greatly benefit from a principled design approach that addresses highly dynamic node states and that aims to optimize complex trade-offs between use of resources, quality-of-service (QoS), and other relevant parameters. We have identified three main design goals for sensor network optimization mechanisms:

1. Flexible: The wide and diverse range of sensor network applications highlights the need for a flexible means to customize network behavior in order to meet each applications' performance goals.
2. Global: The nodes in sensor networks collaborate to achieve a common network-wide goal. As such, sensor network optimization mechanisms should provide global network performance guarantees.
3. Adaptive: The local node states in sensor networks are highly dynamic. An optimization strategy for sensor networks should adapt to changes in local node states.

Sensor network optimization methods should balance the two conflicting design challenges of providing global performance guarantees and of ensuring adaptability to

• R. Jurdak is with the PRISM Laboratory, School of Computer Science and Informatics, Belfield, Dublin 4, Ireland. E-mail: Raja.Jurdak@ucd.ie.

• P. Baldi and C.V. Lopes are with the School of Information and Computer Sciences, University of California, Irvine, Irvine, CA 92697-3435. E-mail: {pfbaldi, lopes}@ics.uci.edu.

Manuscript received 13 Oct. 2005; revised 1 June 2006; accepted 28 Nov. 2006; published online 7 Feb. 2007.

For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-0301-1005. Digital Object Identifier no. 10.1109/TMC.2007.1037.

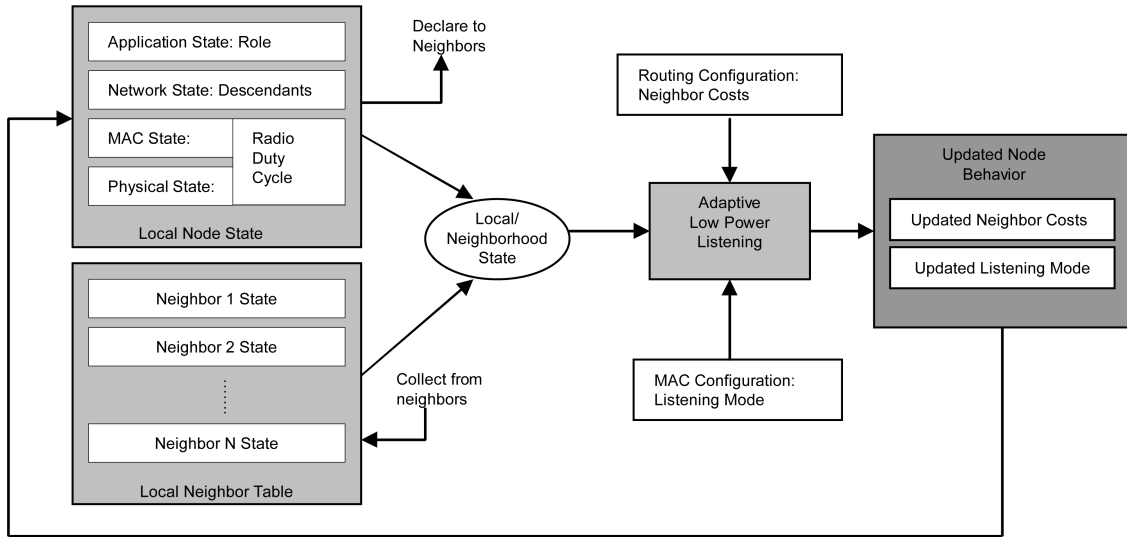


Fig. 1. Cross-layer optimization framework.

local node states. In our optimization framework, we distinguish between three types of state information. Node state is the vector of parameters that represents the cross-layer state of a single sensor node. For a particular node  $N$ , the neighborhood state is the vector of node states of the direct neighbors of  $N$ . The global network state is the vector of all node states in the network.

Optimizing network behavior is trivial if each node has the global network state. However, in multihop sensor networks, having a global network state at each node is not a scalable solution, since it involves excessive communication overhead as the network size grows. To address these challenges, several previous works have proposed greedy locally optimal decisions [1], [2] based on 1-hop neighbor states to make on-demand routing decisions. In our framework, we also enable nodes to greedily optimize their behavior based on the local state and neighborhood state, but we adopt a proactive collaboration strategy. Our choice for a proactive routing strategy stems from the time-driven nature of our target application, so nodes can easily maintain their neighbors' states by piggybacking state information in periodic routing messages, minimizing overhead communication. This collaboration strategy strikes a balance between the two seemingly contradicting goals of providing network-wide performance guarantees and ensuring adaptability to local sensor node states.

## 1.2 Optimization Framework

Within the rationale of global optimization through greedy local decisions, we consider our general sensor network optimization framework from [6], which adopts a flexible node state representation consolidating state variables from several layers of the communication stack. The work in [6] has applied the framework to three diverse ad hoc and sensor network application scenarios with different communication technologies. In this paper, we tailor the framework for a monitoring sensor network application, as shown in Fig. 1. At the routing layer, the framework enables nodes to set their neighbors' routing cost according to the neighborhood state. The flexible and cross-layer state

representation ensures that routing behavior in the network adapts to all relevant state parameters.

At the MAC layer, nodes can also adapt their behavior according to the cross-layer state representation. Radio power consumption is one of the main causes of power consumption at sensor nodes. In monitoring networks, idle listening is the main contributor to radio power consumption. In order to minimize idle listening, each node can adapt its radio duty cycle according to its current local and neighborhood states.

## 1.3 Adaptive Mechanisms

Enabling individual nodes to adjust their routing and MAC behavior according to their local state requires mechanisms that are adaptive, flexible, and modular. Fig. 2 shows the communication mechanisms that drive the optimizations in this paper. At the network layer, our model proposes a flexible cost function for routing optimizations, inspired by the work of Baldi et al. [7]. In this paper, the cost function metrics include power terms related to the duty cycle of node radios and sensor activity, although additional QoS metrics, such as delay or reliability, can be included too. Minimization of the cost function determines the routing strategies of the network.

At the MAC layer, to validate our framework, we choose BMAC [8], a modular and flexible sensor network MAC protocol which aims at reducing idle listening at sensor nodes. BMAC provides interfaces that enable services and applications to set low power listening modes and transmit modes on a per-packet basis if needed. The creators of BMAC also suggest that further power savings could be achieved by using the interfaces to set listening and

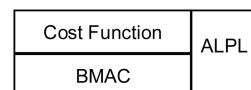


Fig. 2. Communication mechanisms.

transmit modes according to additional information on the application and operation of a sensor network.

Building on BMAC, we had proposed in earlier work [9], [10] a cross-layer mechanism called Adaptive Low Power Listening (ALPL) to provide a seamless basis for locally setting the listening mode while ensuring reliable data delivery. ALPL also ensures consistency for joint optimization at the MAC and routing layers. In the original BMAC protocol, setting a network-wide listening mode disregards the nonuniform and dynamic local states of individual nodes. Enabling each node to set its own listening mode according to its local state is more energy-efficient, but it is difficult to predict the state of each node prior to deployment. To address these challenges, ALPL supports the adaptation of listening modes in BMAC to local sensor node states, and it enables a node to learn the listening mode of its neighbors in order to ensure correct data delivery.

Our earlier versions of ALPL have represented node state as a combination of two aspects: number of descendants in the routing tree and radio duty cycle. A node's number of descendants is a network layer aspect that indicates the node's forwarding load. For instance, leaf nodes have no forwarding load. The radio duty cycle incorporates knowledge about the duty cycles of neighbors into the decision of optimal listening mode, quantifying how busy a node has been in the recent time window. The benefits of the ALPL's state representations were evaluated through deployment experiments for a time-driven monitoring sensor network test-bed.

In this paper, we expand the node state representation to include the role of each node. Certain events or queries may trigger nodes to assume different roles during deployment, causing increased resource usage at these nodes. Including role information into node state representation accounts for heterogeneous roles and enables nodes to better adapt their listening mode and routing costs to dynamic network conditions. To assess the benefits of the expanded state representation, this article presents four new sensor network test-bed deployment experiments for an event-driven network application. This article also highlights the cross-layer design aspects of earlier ALPL optimizations, and it presents our tailored cross-layer framework for this purpose. Finally, this article revisits earlier ALPL results with a clearer representation and a more involved analysis of stability issues.

## 1.4 Overview of the Paper

In sum, the novel contributions in this paper are threefold:

- formalization and tailoring of a flexible optimization framework for sensor networks,
- expansion of previous ALPL state representations within the framework and analysis of the resulting energy benefits, and
- validation of the framework through experiments on a test-bed of sensor nodes running time-driven and event-driven applications and evaluation of the framework's energy benefits on a global network basis and on a node-specific basis

The rest of the paper is organized as follows: Section 2 discusses related work, including BMAC. Section 3 presents

our cross-layer optimization framework and its building blocks. Section 5 presents our experiments to validate the approach on mica2 motes. Section 6 discusses the results and concludes the paper.

## 2 RELATED WORK

### 2.1 Cross-Layer Approaches

Several previous models [3], [4], [5] have adopted a cross-layer design approach for sensor networks. Most of these models propose cross-layer optimizations involving specific mechanisms, such as optimal power control at the physical layer, link scheduling at the MAC layer, routing flow at the network layer, and QoS constraints at the transport layer. Our model generalizes these approaches by adopting a flexible cross-layer representation of node states which enables any combination of cross-layer variables, including application state variables, to affect network behavior.

Woo and Culler [11] propose another cross-layer scheme that couples transmission control with media access. Their approach enables nodes to adapt their data origination rate according to their position within the routing tree in order to promote fairness among downstream and upstream flows. ALPL also considers a node's location within the routing tree, along with duty cycle and role information, in order to choose the node's listening mode that optimizes local energy consumption.

Another cross-layer proposal by Culler et al. [12] outlines the guidelines for establishing a sensor network architecture that enables interoperability among different components. Their recommendation is to have a Sensor Protocol (SP) abstraction layer, similar to IP in the Internet, over which all new sensor network protocols and services could reside. They also propose cross-layer visibility and management of several aspects, such as power and security. ALPL takes a step in this direction by providing higher layer services with a seamless platform for setting listening modes in BMAC according to their requirements.

Within a proposed implementation of the SP abstraction layer, Polastre et al. [13] present an adaptive mechanism for adjusting the preamble length in BMAC. They propose the infrequent transmission of packets with long preambles to enable neighboring nodes to learn each other's wake-up schedules. Nodes can then send packets with short preambles to communicate with their neighbors, thereby reducing BMAC's preamble overhead. Broadcast, discovery, and retransmitted packets still use long preambles for more aggressive transmissions. ALPL [10] independently proposes setting preambles according to the intended receiver's current listening mode, which is determined by the node's cross-layer local and neighborhood state information.

### 2.2 Local State

A central challenge in sensor networks is that no node has a global view of the entire network. Romer and Mattern [14] examine the effectiveness of event-notification mechanisms in supporting the detection of real events in sensor networks by communicating the local state changes in individual nodes. Most event-driven mechanisms are reactive in nature. Liu et al. [15] realize the need for

collaboration and state-sharing among sensor nodes in order to achieve application requirements. They propose collaboration among groups of nodes with a flexible grouping strategy depending on application needs. Here, we consider a proactive collaboration among sensor nodes in the same communication neighborhood in the form of state information sharing within periodic routing messages.

### 2.3 Cost Optimization

Previous efforts [7], [16] have also realized the need for optimizing global cost in wireless networks, which determines the routing strategies of the network. Baldi et al. [7] develop a global cost function for wireless ultra wideband radio networks, based on the cost of individual links. Their cost function is additive and considers transmission power, link setup cost, interference, delay, and reliability. Mhatre et al. [16] optimize the hardware cost of heterogeneous sensor networks with a lifetime constraint. Our framework focuses on dynamic cost optimization, so it disregards the hardware cost which is static once the network is deployed.

### 2.4 Energy-Efficiency

In sensor networks, energy efficiency and load balancing are the primary metrics of interest. Many protocols have been designed to provide energy-efficient behavior at both the MAC layer [17], [19] and the routing layer [21].

#### 2.4.1 Routing Protocols

Sensor network routing protocols are either proactive [22], [23] or reactive [24], [25]. Proactive protocols maintain network or neighborhood routing state tables at each node. Reactive protocols compute optimal routes on-demand. Proactive protocols are advantageous for networks that require limited mobility, low latency, or high throughput. In contrast, reactive protocols are suitable for high mobility, low throughput, high latency networks. In this paper, we consider a typical stationary time-driven monitoring sensor network in which the periodic nature of data transmission fits well with periodic state exchanges in proactive routing protocols. As a result, we consider a proactive and periodic routing protocol in which nodes store the routing state of their one-hop neighbors.

Although many cost metrics have been proposed for routing in sensor networks, we focus the discussion here on metrics for optimizing energy consumption. Previous work has examined energy optimizing routing strategies, where cost metrics include residual battery energy [26], [27]. The residual battery capacity strategy is intuitively valid, but the discharge of real batteries becomes nonlinear or unpredictable at a certain voltage level, which effectively cancels battery considerations from routing decisions at some point during the deployment [28]. In our work, we consider the radio, sensor, and processor duty cycles in the recent time window as the energy metric for routing. We believe that this metric is more expressive of each sensor node's energy profile since it is independent of the battery technology and it does not rely on unpredictable battery discharge models.

#### 2.4.2 MAC Protocols

At the MAC layer, idle listening constitutes a large portion of power consumption because data is sent infrequently.

This effect is even more pronounced in monitoring sensor networks [29]. Thus, energy-efficient MAC protocol proposals have focused on minimizing idle listening at sensor nodes.

IEEE 802.15.4 [18] is a standard with physical and MAC layer specifications for low rate, low power, short range networks, including sensor networks. IEEE 802.15.4 specifies a plethora of functionality choices, many of which may never be used. As a result, several researchers have proposed new MAC protocols on top of the 802.15.4 PHY layer.

SMAC [30] is a heavyweight MAC protocol for sensor networks that relies on time synchronization and scheduling among nodes to enforce periodic sleep and listen schedules. SMAC reduces energy consumption and provides scalability at the cost of per-hop fairness, throughput, and latency. A more recent version of SMAC [32] has introduced adaptive duty cycles by enabling a node to snoop on neighbors' RTS and CTS messages in order to schedule its own wake up time. Because nodes have to maintain neighbors' schedules, SMAC is not sufficiently scalable for large-scale networks of resource-limited nodes.

T-MAC [31] also proposes adaptive duty cycles to address the nonuniform traffic patterns in sensor networks. T-MAC is similar to SMAC in its essence, but it introduces early sleeping to enable nodes that are scheduled to be active to go into sleep mode if they are idle. T-MAC suffers from similar complexity and scaling problems as S-MAC because it trades off a short active time for reduced adaptivity to changing network conditions.

El-Hoiydi et al. propose Wireless Sensor MAC (WiSeMAC) [20], an asynchronous MAC protocol that relies on the preamble sampling technique. Sensor nodes periodically sample the channel for activity. Packets are sent with preambles of equal duration to the channel sampling period to ensure that the receiver will detect the packets. Although WiSeMAC nodes wake up with the same period, the individual node offsets are independent. To address the inherent trade-off between listening power consumption (for period channel sampling) and transmission power (due to preamble size), WiSeMAC proposes a mechanism for a node to learn the sampling schedule of its neighbors in order to send packets with short preambles during the intended receiver's wake-up time.

The recent similar work by Polastre et al. independently proposes BMAC [8], an asynchronous and lightweight sensor network MAC protocol that aims at providing versatile medium access while keeping the MAC functionality as simple as possible. Because it is an asynchronous protocol, BMAC eliminates the communication and processing overhead for scheduling and synchronization, which reduces energy consumption. BMAC enables each node to wake up periodically to check for channel activity. The wake-up period is referred to as the check interval. BMAC defines eight check intervals, and each check interval corresponds to one of BMAC's eight listening modes. To ensure that all packets are heard by the nodes, packets are sent with a preamble whose reception time is longer than the check interval. BMAC therefore defines eight different preamble lengths, referred to as transmit

modes. Additionally, Polastre et al. analytically derive optimal listening modes based on the number of neighbors of a node. In their experiments, they determine the maximum neighborhood size in the network, and they set the optimal listening mode for that neighborhood size. The experimental results yield significant energy savings for BMAC over previous protocols, such as SMAC. BMAC is also the standard MAC protocol in the communication stack of TinyOS [35], the primary sensor network research platform.

As in T-MAC and SMAC, our paper addresses the nonuniform node states by adapting listening modes in BMAC to enable adaptive duty cycles. We choose BMAC because of its flexibility and scalability, which are two of our main design goals. Instead of proposing a new MAC protocol, this paper's focus is designing a mechanism (namely, ALPL) that enables each node to adapt its duty cycle based on flexible cross-layer node state representations.

### 3 CROSS-LAYER OPTIMIZATION FRAMEWORK

The algorithm that characterizes our framework periodically runs the same three basic steps for any network scenario: 1) gather neighborhood state information, 2) perform local calculations on gathered state, and 3) modify local configuration accordingly.

The details of the algorithm and its implementation are highly dependent on the network scenario. As such, this section discusses the details of the algorithm tailored for a monitoring sensor network application. We begin by introducing the motivation for ALPL within the framework. Next, we describe the node interaction that enables nodes to set listening modes through ALPL and to exchange their state information. We subsequently describe the three state representations that are considered in this paper. Next, we explain how each node uses its neighborhood state information to calculate the routing cost of each neighbor through the cost function. Finally, we present the routing issues to ensure correct data delivery and route stability with ALPL.

#### 3.1 Adaptive Low Power Listening

Adaptive Low Power Listening [9], [10] is a cross-layer mechanism that adapts the listening mode at each node according to its local state, while ensuring correct data delivery.

The motivation for ALPL is to address the unpredictable and dynamic node states in sensor networks which are affected by factors such as interference variations and dynamic node membership. Network designers have to make conservative assumptions in determining the network configuration. In the case of energy optimization, conservative assumptions lead to setting a network-wide listening mode in BMAC prior to network deployment. This causes unnecessary idle listening to occur in less active portions of the network. ALPL's purpose is to reduce idle listening in BMAC by allowing each node to set its own listening mode depending on its local node and neighborhood states. The rationale is that, in dynamic sensor networks, each node always has the most up-to-date view of its own local state [34]. Node states can be defined by the network designer or

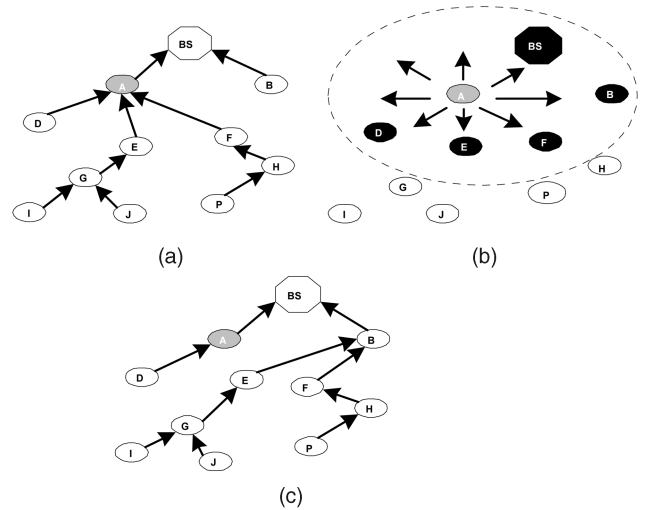


Fig. 3. (a) Nodes form their routing tree. (b) Each node periodically announces its current state, including its listening mode and battery state, enabling neighbors to use the appropriate transmit mode. (c) A high duty cycle at  $A$  causes neighbors to increase  $A$ 's routing cost and choose a new parent  $B$ . Node  $A$  listens less often as it has fewer packets to forward.

operator depending on the applications' goals and quality of service requirements.

#### 3.2 Node Collaboration

The collaboration strategy in our framework enables a node to learn its neighborhood state information, to set its own listening mode accordingly, and to adapt its transmit mode to fit the listening mode of its routing parent. We note here that ALPL requires no synchronization among node clocks since it builds on BMAC listening modes. The *only* requirement for correct communication is *proper matching* of the *sender's preamble length* and the *receiver's listening mode*. Our deployment experiments in Section 5 evaluate the soundness of this approach through observation of long-term data delivery.

Fig. 3 illustrates the node interaction to enable nodes to set their listening mode adaptively. We assume a proactive routing protocol in which nodes periodically send routing update messages to declare their routing information to their neighbors.

Initially, nodes are unaware of their neighborhood state, so all nodes listen at an initial listening mode  $L_{init}$  and use the corresponding transmit mode  $T_{init}$ , both of which are known a priori to all nodes. Each node begins sending periodic route update messages to declare its presence and state. Once nodes learn of their neighbors' presence, a routing graph is formed and data flows toward the base station (Fig. 3a). As a result, each node learns the state of its direct neighbors. Before sending the next route update message, a node  $A$  first sets the optimal listening mode  $L_A$  for its local state. Then, node  $A$  sends a routing update message that includes its new listening mode and state information along with other routing information (Fig. 3b).

All of  $A$ 's neighbors hear the routing update message, and they learn  $A$ 's current listening mode  $L_A$  and  $A$ 's state information. Each neighbor of  $A$  records  $A$ 's listening mode and state information in its local neighbor table.

Consequently, each node in the network always has up-to-date information on the state of its neighbors. Whenever a node  $D$  chooses  $A$  as a routing parent, it simply checks its neighbor table for  $A$ 's listening mode  $L_A$ .  $D$  then sends its data packets using the transmit mode  $T_A$  that matches  $L_A$ . Similarly, nodes that receive a routing update message from their current parent indicating that the parent has a new listening mode adapt their transmit mode accordingly.

For a particular node in ALPL, the only concern for data delivery is whether the routing parent is reachable. If so, then the node can transmit with the proper preamble length. Otherwise, the node detects the unreachability of the routing parent through missed route update packets and attempts to find another suitable routing parent. If the node hears a routing update packet from the original routing parent at some later point, the node can rerun the algorithms for choosing the routing parent and the appropriate listening mode.

### 3.3 State Representations

Our state representations consist of three aspects: 1) number of descendants, 2) duty cycle, and 3) role. In this section, we describe each of these three aspects in detail.

#### 3.3.1 Number of Descendants

Our first representation of state considers a node's number of descendants in the routing tree, which has significant impact on a node's energy profile in data gathering and monitoring applications. In monitoring applications, data flow is typically toward a single data sink. The basic requirement for correct data delivery is for each node to listen often enough to hear all the packets that it must forward toward the data sink. The number of packets that a node forwards depends on the number of its descendants in the routing tree.

In order to determine its optimal listening mode, each node  $N$  learns how many descendants it has in the routing tree by counting the number of packets  $\gamma$  that it forwards during a route update interval. The number  $\gamma$  indicates how busy  $N$  was during the last interval. When it is time to send the next routing update message,  $N$  first sets its listening mode to the optimal listening mode  $L_N$  for a traffic load of  $\gamma$  packets. Then,  $N$  sends a routing update message that contains  $L_N$  along with other routing information.

Including the number of descendants into ALPL's state representation optimizes the overall power consumption in the network. Each node optimizes its local power consumption through the selection of an optimal listening for its current forwarding load. The distributed optimization of local node power consumption leads to an optimization of network power consumption. The number of descendants only exploits the load imbalance in the network to opportunistically optimize power consumption at lightly loaded nodes. The next two sections present additional node state aspects that promote load balancing.

#### 3.3.2 Duty Cycle

Our second state representation combines the number of descendants and the node's duty cycle. A high radio duty cycle indicates that the node's radio has been highly active up till the present point in time and vice versa. Given the

inherently nonuniform energy consumption in sensor networks, enabling nodes to adapt their behavior according to their radio duty cycle can help balance the power consumption in the network.

Sharing duty cycle information among nodes involves minimal overhead communication. Each node can piggyback its duty cycle value within its routing update messages in order to declare its power state to neighbors. As a result, nodes learn the radio duty cycles of all their one-hop neighbors and store this information in their local neighbor table.

Including the radio duty cycle into ALPL's state representation enables local load balancing of network traffic. Each node can consider and compare the recent activity level of neighbor transceivers. Neighbors with a relatively higher radio duty cycle are penalized through higher routing cost, which diverts traffic away from busier nodes to other nodes to achieve more uniform energy consumption. The radio duty cycle promotes load balancing to the extent allowed by the routing graph. As long as the routing graph provides a node with more than one alternative for a routing parent, the radio duty cycle information contributes favorably to load balancing.

As an example, consider Fig. 3 again. The routing tree in Fig. 3a puts most of the forwarding burden on node  $A$ . As a result,  $A$  depletes its battery resources quicker than node  $B$ . In order to shift its forwarding load,  $A$  declares its high duty cycle to its neighbors, causing neighbors to increase  $A$ 's routing cost. This in turn causes most of  $A$ 's current children to choose another parent whenever possible (Fig. 3c). Having diverted most of its forwarding load to node  $B$ , node  $A$  begins listening with a longer check interval to reduce its listening power consumption.

#### 3.3.3 Role

The nonuniform state of sensor nodes also stems from the roles that the nodes may take during deployment. The number of descendants is a routing aspect and it represents the present state of the node. The duty cycle is a physical and MAC layer aspect that represents the past state of the node. The node's role provides information on application functionality and represents its projected state into the future. In the context of our energy optimization study, the node's role can play a part in determining the node's optimal listening mode and neighbors' routing costs. The rationale is to incorporate additional knowledge about the expected power profile of a node's role in optimal local decisions.

In order to integrate role into network decisions, nodes should share their role status with their neighbors. Sharing role information among nodes involves minimal overhead communication. For example, in a network where nodes may take one of two roles, each node can include its role status within its routing update messages through a single bit. As a result, nodes learn the roles of all their one-hop neighbors and store this information in their local neighbor table.

As in the case of radio duty cycles, including node roles into ALPL's state representation promotes local load balancing of network traffic. Each node can consider and compare the recent sensing activity level of neighboring

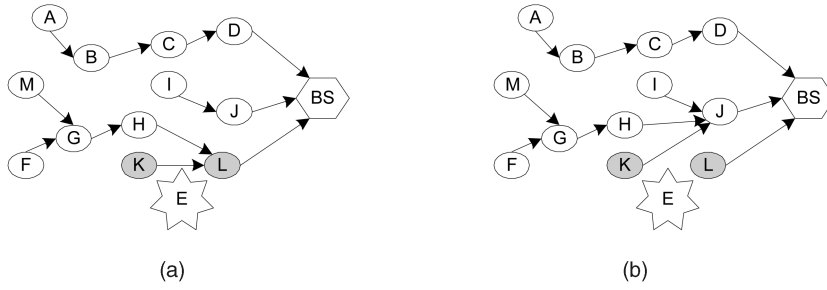


Fig. 4. An event-driven tracking sensor network.

nodes. Thus, a node can divert traffic away from nodes with high sensing activity, which contributes to load balancing. In homogeneous networks where all nodes behave the same, the role variable does not contribute to load balancing. However, in event or query-driven networks, certain nodes may assume more active sensing roles during the deployment. In these cases, the power consumption for sensing becomes more significant and it should be considered for load balancing considerations.

Consider the network in Fig. 4. While nodes are periodically sending their data with period  $T$ , much of the forwarding load is focused at node  $L$ . At some point during the deployment, an event  $E$  occurs at the lower periphery of the network. Nodes  $K$  and  $L$  detect event  $E$ , switch their role to tracker, and begin sensing and reporting data at  $T/2$ . The neighbors of  $K$  and  $L$  learn that these two nodes are now tracker nodes and thereby increase the routing cost of these two nodes. Fig. 4b shows the resulting topology. Nodes  $K$  and  $H$  choose a new parent, node  $J$ , in order to avoid using the tracker node  $L$  as a forwarder. The topology in Fig. 4b puts most of the forwarding load on node  $J$ . The energy consumption resulting from the high forwarding load of node  $J$  is balanced by the energy consumption for more frequent sensing and data reporting at node  $L$ .

### 3.4 Cost Function

Our primary platform for sensor network development is TinyOS [35], developed at UC Berkeley. Within TinyOS, the standard routing protocol is called MintRoute [36]. MintRoute is a proactive routing protocol in which nodes send periodic routing messages to declare their local states to their 1-hop neighbors.

In all proactive routing protocols and particularly in MintRoute, each node periodically selects its routing parent. A node  $N$  first selects the highest contender  $M$  in its neighbor table with the least routing cost at the current time. Next,  $N$  compares the cost of  $M$  with the cost of the current routing parent  $RP$ .  $N$  chooses  $M$  as its new parent only if

$$C_{mint}(M) + \epsilon < C_{mint}(RP), \quad (1)$$

where  $C_{mint}(N_j)$  is the routing cost of node  $N_j$  and  $\epsilon$  is the switching threshold that ensures that a node switches its routing parent only when there is an appreciable benefit in doing so.

The original cost metric in MintRoute is the Expected Transmission Count (ETX), which assigns the link weight as

the product of the reciprocal of the forward and backward link qualities. Minroute then uses a shortest path algorithm for determining the best route to a destination. Nodes that employ the ETX routing metric snoop on the neighborhood links to keep track of the forward and reverse delivery ratio on each link. The ETX metric in MintRoute implicitly minimizes radio power consumption. Since our framework aims at making routing decisions dependent on the node energy states, we introduce additional cost metrics that are direct functions of the aspects that constitute node state.

#### 3.4.1 Duty Cycle Awareness

The number of descendants aspect is considered for setting local listening modes rather than setting neighbor costs, so this aspect does not require an explicit routing cost metric. For the duty cycle aspect, we introduce the cost metric  $C(\text{radio})$ , which denotes how busy a particular neighbor's radio is relative to other neighboring nodes. Highly loaded nodes should have a higher routing cost. Therefore, we express  $C(\text{radio})$  for a neighbor  $N_j$  as

$$C(\text{radio}) = \epsilon \frac{\delta_j - \sum_{i=0}^k \delta_i/k}{\sqrt{\sum_{i=0}^k \delta_i^2/k - \left(\sum_{i=0}^k \delta_i/k\right)^2}}, \quad (2)$$

where  $k$  is the number of neighbors in the node's local neighbor table and  $\delta_i$  is the radio duty cycle of neighbor  $N_i$ . Equation (2) compares the radio duty cycle of neighbor  $N_j$  to the average duty cycle in the neighborhood during the recent time window and normalizes the difference. The normalized difference determines the extent of statistical deviation of the radio activity of  $N_j$  among all the nodes in the neighborhood.

#### 3.4.2 Role Awareness

The duty cycle is a sufficient indicator of energy profiles in cases where the nodes all have the same sensing and processing patterns throughout the lifetime of the network. In many event-driven or demand-driven sensor networks, some nodes change their application behavior, such as the sensing frequency or buffering strategy, based on events in the network. Such cases require the inclusion of information on the current roles of nodes in routing decisions.

The actual cost dependence on the node's role is highly application-specific. In this paper, we consider an event-driven network that specifies that nodes can have one of two sensing frequencies,  $f_1$  or  $f_2$ , where  $f_2$  is double  $f_1$ . A

node uses the following equation to evaluate the sensing cost of a neighbor  $N_j$ :

$$C(\text{sensing}) = \begin{cases} \epsilon \frac{\theta_j - \sum_{i=0}^k \theta_i/k}{\sqrt{\sum_{i=0}^k \theta_i^2/k - (\sum_{i=0}^k \theta_i/k)^2}} & \theta_j = 1 \\ 0 & \theta_j = 0, \end{cases} \quad (3)$$

where  $k$  is the number of neighbors in the node's local table.  $\theta_i$  is a Boolean variable that takes the value of 1 if  $N_i$  is sensing data at frequency  $f_2$  (tracker node) and takes the value of 0 if  $N_i$  is sensing data at frequency  $f_1$ . Equation (3) compares the sensing activity of neighbor  $N_j$  to the average sensing activity in the neighborhood during the recent time window and normalizes the difference. The normalized difference determines the extent of statistical deviation of the sensing activity of  $N_j$  among all the nodes in the neighborhood.

### 3.4.3 Overall Cost Metric

The overall routing cost should strike a balance between the need for correct and timely data delivery on one hand and uniform energy cost on the other. Furthermore,  $C(\text{radio})$  and  $C(\text{sensing})$  should play a role in determining the routing parent only if the node  $M$  is at the same or at a lower level than the current routing parent  $RP$ .

Therefore, the new overall cost of a neighbor includes the original MintRoute routing metric, as well as the power cost according to the following equation:

$$C(M) = \begin{cases} C_{\text{mint}}(M) + \alpha C(\text{radio}) + \beta C(\text{sensing}) & H(M) \leq H(RP) \\ C_{\text{mint}}(M) & H(M) > H(RP), \end{cases} \quad (4)$$

where  $\alpha$  and  $\beta$  are constants representing the weights of  $C(\text{radio})$  and  $C(\text{sensing})$ , respectively.  $H(N)$  indicates the hop count of node  $N$  from the base station. With the new cost definition, neighbors with higher power consumption (due either to radio duty cycle or to increased sensing activity) have a higher routing cost, and they are less likely to be chosen as forwarders.

### 3.4.4 Network Cost

The network cost in our framework is the sum of the costs of individual nodes [7]:

$$NC = \sum_N C(N). \quad (5)$$

Since optimizing  $NC$  is not feasible, each node selects its routing parent as the node with the least routing cost  $C(N)$ .

## 3.5 Routing Issues

Enabling each node to set its own listening mode adapts MAC protocol behavior to the node's state. The MAC protocol adaptation should be accompanied with routing protocol adaptation in order to maximize performance gains.

The concept of adaptive listening modes raises the possibility that some nodes may not hear the packets sent

by their neighbors because of mismatched preamble lengths and check intervals. For example, if a node A sends a packet with a short preamble to its parent B, one of node A's neighbors C that is listening infrequently may miss node A's packet. This situation does not affect data delivery, since it is only necessary for A's parent B to hear the packet. Missing a routing update packet is more detrimental, since routing packets hold important information on neighborhood routing state changes.

We implement modifications to MintRoute to address missed routing update packets. A central issue in designing ALPL is to ensure that asymmetric listening modes do not affect maintaining an up-to-date neighborhood view at each node. Achieving this goal requires that nodes always hear the routing update packets of their neighbors. Thus, ALPL specifies that nodes always send their routing update packets with the longest preamble, so that a neighbor in any listening mode can hear the update packets. Secondly, because MintRoute determines link quality by snooping on forward and backward links, we modify MintRoute so that nodes only snoop on periodic routing updates instead of data packets to determine the link quality to their neighbors. Monitoring route update packets for determining link quality ensures that asymmetric listening modes at neighboring nodes have no detrimental effect on link quality, because all routing update packets are sent with the longest preamble.

The current implementation of ALPL piggybacks state information on routing update messages, which closely couples the period of the state information updates with routing updates. We view this coupling as favorable, since routing protocols that send frequent updates target highly dynamic networks, where frequent state information updates are also required. In our current implementation of ALPL, the default routing update period is set to half of the data sampling period.

The longer preambles used for routing update messages represent the main cause of energy overhead of ALPL. For applications with data sampling periods in the order of minutes, the control overhead is relatively low as routing update messages with long preambles are sent infrequently. The increase in control overhead is counterbalanced by the reduction of idle listening power, since it is the idle listening power consumption that dominates the overall power consumption at a node in most monitoring applications.

Because state information is sent infrequently, nodes may transiently have stale state information about the listening mode of their parents due to missed route update packets. If the parent is still alive and within range, the stale information persists until the time of the next route update message (which is two data sampling periods in our current implementation). During that time, the mismatch between the node's preamble length and the parent's listening mode affects data delivery only if the parent has switched to a less frequent listening mode. In this case, the child node does not receive acknowledgements for its data packet and it realizes that the parent node has not received the packet. If the child node sends the data packet again with no acknowledgement, it tries to send the packet with the

longest preamble in order to aggressively reach the parent node [13], [20]. If there is still no acknowledgment after the third attempt, then the child node gives up and looks for another routing parent within its routing table.

Two factors determine the stability of the new cost metric: 1) the switching threshold  $\epsilon$  and 2) the time-averaging of the cost components. Suppose that a node has two potential routing parents and that the overall routing cost of the two parents is close in value. The node does not switch its current parent until the other candidate's routing cost is less than the current parent's cost by at least  $\epsilon$ , whose value can be tuned for different scenarios. Another issue is route flaps. If  $C(\text{power})$  and  $C(\text{role})$  are computed for a short time window into the past, then one might expect a node to keep switching between the two potential parents. Once it switches to a new parent, the overall cost of the other original parent drops and the node switches back to the original parent during the next period. However, setting a sufficiently large time window into the past, during which  $C(\text{power})$  and  $C(\text{role})$  are computed, ensures that the overall cost metric reflects the aggregated energy consumption of a node so far and avoids dependence of routing decisions on transient changes. The optimal value of  $\epsilon$  and the optimal length of the time window over which the values are averaged are dependent on how dynamic the target network scenario is, and they remain open issues for further research.

## 4 QUALITATIVE ANALYSIS

This section presents the analytical basis for using greedy local decisions to reduce global network power consumption. We assume that the sensor nodes collect sensor data and transmit the data in a packet once in every period  $T$ . The following equation governs power consumption  $E$  at a sensor node [8]:

$$E = E_t + E_r + E_d + E_{listen} + E_{sleep}, \quad (6)$$

where  $E_t$  is the power spent on transmissions during time  $T$ ,  $E_r$  is the power for packet reception during time  $T$ ,  $E_d$  is the power required to collect sensor values,  $E_{listen}$  is the power consumed for checking the channel for activity, and  $E_{sleep}$  is the power consumed while the node is asleep. The quantitative expressions for each energy component are given in [8]. We limit the discussion here to the qualitative aspects that are relevant to ALPL.

In monitoring and data gathering applications, the sampling period  $T$  is typically in the order of minutes. Therefore, each node collects sensor data, transmits packets, and receives packets once every few minutes. On the other hand, nodes wake up to monitor the channel for activity much more frequently, for instance, once every several milliseconds. Thus, idle listening on the channel has a profound effect on the overall power consumption, so reducing idle listening yields significant power savings.

### 4.1 Topology

The aim of the number of descendants metric is to minimize the overall power consumption in the network through local optimization decisions. Each node can optimize its own power consumption  $E$  locally by selecting its own

optimal listening mode while maintaining correct and timely data delivery. Minimizing  $E$  on a per-node basis reduces the overall network power consumption and builds on the following observations about (6):

1.  $E_d$  and  $E_{sleep}$  are not significant factors in determining optimal listening mode in a homogeneous monitoring network.  $E_d$  is equal for all nodes.  $E_{sleep}$  is at least an order of magnitude smaller than the other terms in (6), so it has a negligible effect on  $E$ .
2.  $E_t$  and  $E_r$  depend on the node's position in the logical topology. If a node is a leaf in the routing tree, it has fewer packets to forward.
3. The listening mode at node  $N$  determines  $E_{listen}$ . It also determines the preamble length for packets that are received at  $N$ . Consequently, the listening mode at  $N$  affects  $E_r$  at  $N$  and  $E_t$  at  $N$ 's children.

For example, a more frequent listening mode at  $N$  increases  $E_{listen}$  and decreases  $E_r$  at  $N$ .  $E_{listen}$  increases because  $N$  wakes up more frequently to check for channel activity, and  $E_r$  decreases because the frequent listening enables  $N$ 's neighbors to send their packets to  $N$  with shorter preambles, thereby reducing packet reception time at  $N$ .

Similarly, the listening mode of  $N$  affects  $E_t$  at the children  $c_i$  of  $N$ . A more frequent listening mode at  $N$  enables  $c_i$  to send its packets with shorter preambles, thus reducing  $E_{t_i}$ . Through similar reasoning, less frequent listening at  $N$  forces  $c_i$  to send packets with long preamble and consume more power for packet transmissions.

These dependencies further support the need for setting per-node listening modes. In practice, each node locally computes  $E_t$ ,  $E_d$ , and  $E_{sleep}$  and then selects the listening mode that provides the combination of  $E_{listen}$  and  $E_r$  that yields the lowest power consumption  $E$ .

Our cross-layer optimization framework builds on the need to reduce idle listening and to balance power consumption. The framework most basic state representation only considers number of descendants in the routing tree, which yields energy savings at nodes with few descendants. We now consider a case study analysis to illustrate the benefits, scalability, and shortcomings of this most basic form of ALPL.

### 4.2 Case Study

To analyze the trade-offs involved in ALPL and to compare the power consumption of ALPL and BMAC, we consider a case study of a static 127-node network with a binary tree topology. Although the topology of an actual sensor network can be both irregular and transient according to environmental conditions as well as location, this case study serves the purpose of validating the analytical basis and the scalability of ALPL. In the next section, we perform experiments with actual sensor nodes to further validate our model in a more dynamic and realistic scenario. Our analysis for this network assumes that the sensor nodes sense the environment and send their data to the base station once every 3 minutes.

In their evaluation of BMAC, Polastre et al. use the following method for assigning a listening mode that favors the busiest node to improve network lifetime:

TABLE 1  
A Comparison of the Listening Modes  
at Each Level in the Network

Level	1	2	3	4	5	6	7
BMAC Check Interval (ms)	10	10	10	10	10	10	10
ALPL Check Interval (ms)	10	20	20	50	100	200	200

- compute the expected number of descendants of the busiest node in the network and
- set the network-wide listening mode to favor the busiest node.

Since the busiest node in the network has the largest forwarding load, this method for setting the listening mode typically means that all the nodes in the network use the higher duty cycle setting which suits the busy node. This gives rise to one of the motivations of ALPL: to enable nodes that are not as busy to choose their appropriate duty cycle in a decentralized manner.

Table 1 compares the check interval between network-wide listening modes (plain BMAC) and ALPL at each level in the 127 node binary tree network. In plain BMAC, the check interval is set to 10 ms for all the nodes in order to accommodate the forwarding load of the busiest node. In ALPL, the busiest node (the node at level 1) selects the same listening mode as BMAC nodes, and the remaining nodes select their own listening modes based on their topology position. The following example illustrates the listening mode selection process in ALPL.

We consider how a level 2 node in the 127-node binary tree selects its optimal check interval in ALPL. A level 2 node sends 63 data packets each update period, where one packet is locally generated and the rest are forwarded packets from its descendants. The level 2 node also uses a preamble length of 28 bytes in order to match its parent's check interval of 10 ms. The resulting transmission power  $E_t$  in ALPL for a level 2 node is 0.5321 mW. Using the fact that it receives 62 data packets from its descendants each update period, the level 2 node then computes its reception power  $E_r$  for each of the eight possible check intervals. It also computes  $E_{listen}$  and  $E_{sleep}$  for each check interval. Finally, the level 2 node selects a check interval of 20 ms, with a reception power  $E_r$  of 0.8617 mW and a listening power  $E_{listen}$  of 0.865 mW, as the optimal check interval that yields the minimum overall power consumption  $E$  of 2.5 mW at the level 2 node.

All other nodes in ALPL use the same process to select the optimal listening mode. As mentioned earlier, the level 1 node selects a check interval of 10 ms as in the BMAC case because it is the busiest node in the network. Nodes at level 2 and at higher levels choose less frequent listening modes because they have a smaller forwarding load than the level 1 node.

We note here that, as the network size grows, more nodes at lower levels of the tree converge to the generic BMAC behavior. For example, if we double the network size of this case study to 256 nodes, nodes at level 2 choose their optimal check interval as 10 ms. Doubling the network

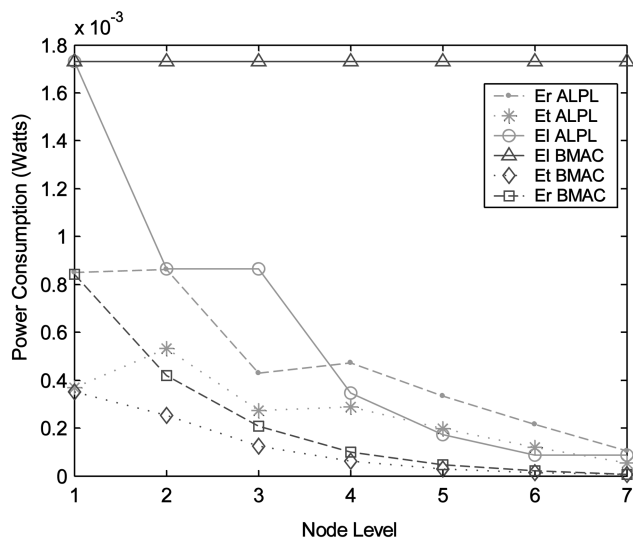


Fig. 5. A comparison of BMAC and ALPL for listening, transmission, and reception power consumption in a sensor network with a binary tree topology.

size further causes level 3 nodes to select the highest duty cycle, and so on.

Fig. 5 compares the sources of energy consumption for the two cases.  $E_d$  is omitted from Fig. 5a because it does not vary with listening mode, and  $E_{sleep}$  is omitted because it has a negligible impact on overall energy consumption. The plots for the listening power consumption in Fig. 5 illustrate the energy savings of choosing per-node check intervals (See Table 1). We observe that ALPL saves more on idle listening power for nodes at higher levels in the tree because these nodes have fewer packets to forward and they can listen to the channel less often. For the level 1 node, the idle listening power consumption is the same for a network-wide listening mode and ALPL because these nodes must listen often enough to accommodate their high forwarding load.

ALPL saves on idle listening power consumption at the cost of increased transmission and reception power, which is an inherent trade-off of the underlying BMAC protocol [8]. The increase in reception and transmission power stems from the use of longer check intervals, which requires long preambles for packets. Note that the level 1 node has the same transmission and reception power in both cases because this node receives and sends packets with the same preamble.

Fig. 6 compares the overall power consumption for BMAC and ALPL on the basis of node levels. For ALPL, power consumption follows a similar trend as the idle listening power consumption in Fig. 5. For plain BMAC, the overall power consumption follows the trend of  $E_r$ , mainly because  $E_{listen}$  is the same for all nodes. Nodes at higher layers exhibit more power savings with ALPL because their low forwarding load enables them to sleep more often.

The lifetime of the network is constrained by the power consumption of level 1 nodes [16], [38]. At first glance, Fig. 6 seems to indicate that the basic form of ALPL does not extend the lifetime of the network despite significant energy savings at all but one level in the tree. However,

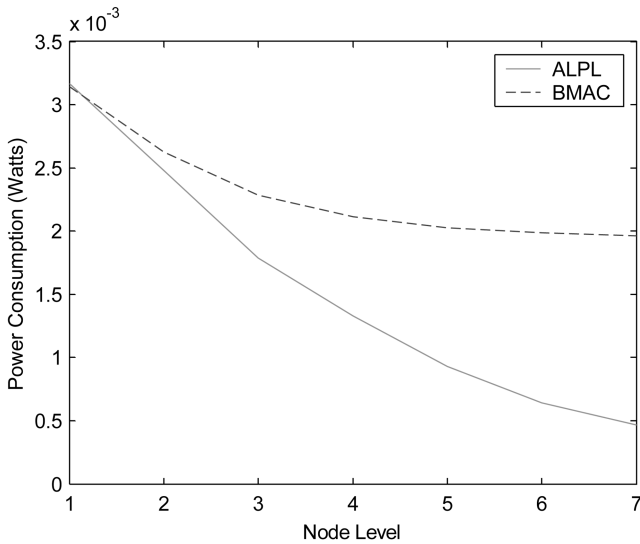


Fig. 6. A comparison of BMAC and ALPL for overall power consumption in a sensor network with a binary tree topology.

recall that sensor networks have a dynamic topology. Varying interference conditions and moving objects typically cause changes in the topology, causing nodes to choose new parents. For instance, a level 2 node may choose the base station as its parent at some point, effectively assuming the role of a level 1 node. Similarly, nodes may choose a parent at a lower level at some point in order to avoid low quality links.

The expanded state representations in our framework yield further energy savings at the most loaded nodes, which contributes to prolonging network lifetime. By having more detailed information about neighbors' states, nodes can more expressively select neighbors' routing costs. As a result, forwarding traffic moves away from loaded nodes and energy consumption becomes more balanced. The next two subsections describe the interaction of duty cycle and role on power consumption in our framework.

### 4.3 Duty Cycle

The energy consumption in sensor networks is nonuniform among the nodes due to communication asymmetries. First, data always flows to one or a few sinks. The nodes that are one hop away from a data sink are called critical nodes [16]. Critical nodes have a larger forwarding burden and consume more energy than nodes further away from the sink [38]. These factors indicate that nodes can use up battery resources at different rates. ALPL can contribute to balancing the power consumption rates among network nodes by manipulating data forwarding patterns and listening modes. Of course, the degree of achievable load balancing depends on the node density and the layout of the nodes.

In terms of (6), highly loaded critical nodes have larger  $E_t$  and  $E_r$ . Because ALPL sets the listening mode according to topology information, a loaded critical node will also choose to listen frequently to the channel, so it has a high  $E_{listen}$ . As a result, the energy consumption  $E$  and the duty cycle at a loaded critical node are higher than that of its neighbors. Through our framework, the loaded critical node

informs its neighbors of its busy state. As a result, the loaded node's children increase its routing cost and choose new parents to forward their packets, thereby reducing its  $E_r$  and  $E_t$  at the loaded critical node. The loaded critical node also switches to a listening mode with a longer check interval to reduce  $E_{listen}$ . The reductions in  $E_r$ ,  $E_t$ , and  $E_{listen}$  yield a significant decrease in  $E$  at the critical node.

### 4.4 Role

In applications where nodes may dynamically select different roles, the sensing power  $E_d$  may become a significant player in determining the node's power consumption relative to its neighbors. For instance, suppose the occurrence of a particular event causes some nodes in the network to double their sampling frequency in order to better track the event. The increased sampling frequency at these nodes increases the sensing power consumption  $E_d$  and it also increases  $E_t$  because the node sends packets more frequently to report the more frequent sensed data. The increase in  $E_t$  falls within the radio duty cycle value so it does not cause any distortion for energy balancing. The more appreciable increase in  $E_d$  has no explicit effect on the radio duty cycle at the node, but it can cause energy imbalances in the network. Including the power consumption  $E_d$  due to the node's role remedies this problem and maintains a balance in overall power consumption  $E$  among the nodes.

## 5 DEPLOYMENT RESULTS

In this section, we investigate the impact of local state-driven optimizations on a testbed of sensor nodes deployed in our laboratory. The sensor nodes in our experiments consist of 14 mica2 motes from Crossbow [39]. Our implementation of ALPL is in NesC [40], a component-oriented variant of C customized for networked embedded systems and built into TinyOS.

The nodes are placed at random positions in the laboratory and the base station is placed near one of the walls of the room. We reduce the transmit power of nodes to limit their radio range, enabling multihop communication. The aim of the experiments is twofold: 1) to assess the effect of state-driven optimizations on the global network power consumption and 2) to evaluate the local node power consumption and energy balancing benefits for state-driven optimizations.

We adopt the method suggested by [8] for computing power consumption. The underlying BMAC design includes several radio states, including active and sleep states. The average current draw for each radio state is fixed. The method employs a counter that keeps track of the time that the radio spends in each power state. Having the current draw and time spent in each state, each node can continually compute its aggregated power consumption so far. The power consumption results shown for our deployments represent the aggregated power consumption for the entire duration of the deployment, including all overhead communication for maintaining routing graphs. To normalize the results, we assign the data point with the highest aggregated power consumption a value of one, and

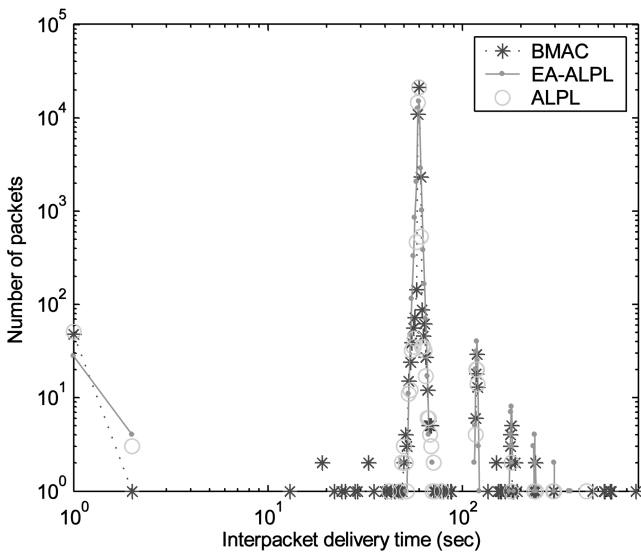


Fig. 7. A comparison of the distribution of interpacket delays arriving at the base station.

we assign the remaining values for that comparison according to that data point.

### 5.1 Time-Driven Sensor Network

The first experiment set considers a time-driven monitoring sensor network with a single data sink. We conduct three experiments for the same physical network topology. In the first experiment, we initially determine the listening mode for the busiest node in the network, and we assign that listening mode in BMAC to all the nodes. In the second experiment, each node runs ALPL and sets its listening mode according to its number of descendants in the routing tree. The third experiment expands the view of local state to include the duty cycle, so nodes run ALPL and select listening modes according to the expanded local state. We refer to this case as Energy Aware ALPL (EA-ALPL).

Each experiment lasts for 43 hours. In all experiments, nodes run the Surge application that is available with the standard distribution of TinyOS. In Surge, nodes sample the sensors and send the data once every minute. Thus, each node sends 2,580 data packets during each experiment. The routing update period for the network-wide listening mode

experiment is 120 seconds. For both ALPL and EA-ALPL experiments, the routing update period is 90 seconds, to allow more adaptive link qualities based on routing update messages. The radio duty cycle weight  $\alpha$  is set to 2 for this experiment set.

#### 5.1.1 Delay and Packet Delivery

We begin by examining the effect of applying ALPL on the packet delivery time to the base station. Fig. 7 compares the interpacket delays arriving at the base station in each of the three experiments. For the majority of packets in all experiments, interpacket delivery time follows a uniform distribution centered at 60 seconds. The deviation of a few seconds for a small number of packets is attributed to clock skews in the nodes. It is worth noting the local peaks at 0 seconds and at constant multiples of the update period. An interpacket delivery time close to 0 seconds indicates a packet retransmission. The number of packet retransmissions is about the same for the three experiments. This result confirms that ALPL does not increase the number of retransmissions over BMAC. The other local peaks represent missed packets. An interpacket arrival time of 120 seconds from a particular node to the base station indicates a single missed packet from that node. Through the same logic, an interpacket arrival time of  $AT$  seconds indicates  $(AT - 60)/60$  consecutive missed packets. The number of missed packets for the three experiments are of the same order, indicating that ALPL does not cause additional packet losses over BMAC. In sum, the results in Fig. 7 have shown that the introduction of ALPL does not affect delay or packet delivery in the network when compared to BMAC.

#### 5.1.2 Overall Power Consumption

The average data yield for BMAC, ALPL, and EA-ALPL is about 98.5 percent. Because the data yield is the same with or without state-driven optimizations, we focus our analysis here on power consumption issues.

Fig. 8a plots the average global network energy consumption for BMAC, ALPL, and EA-ALPL. Both ALPL and EA-ALPL reduce global energy consumption by about 35 percent on average during the deployment. The reduction in global energy consumption stems from the optimal local decisions at each node. In particular, the

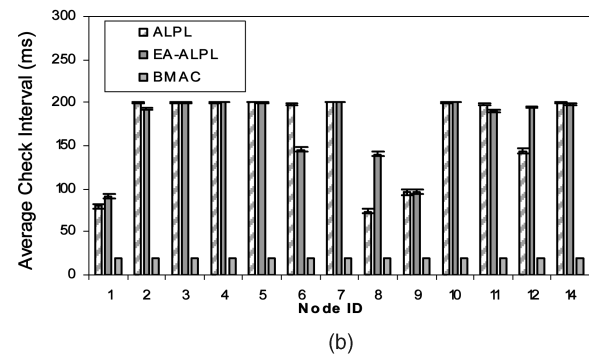
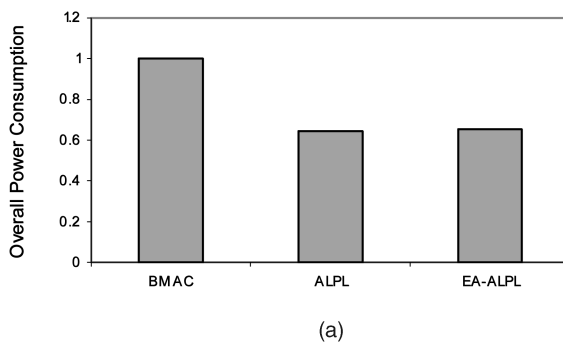


Fig. 8. (a) Global network power consumption during deployment. (b) Average check intervals at each node. The error bars indicate the 99 percent confidence interval.

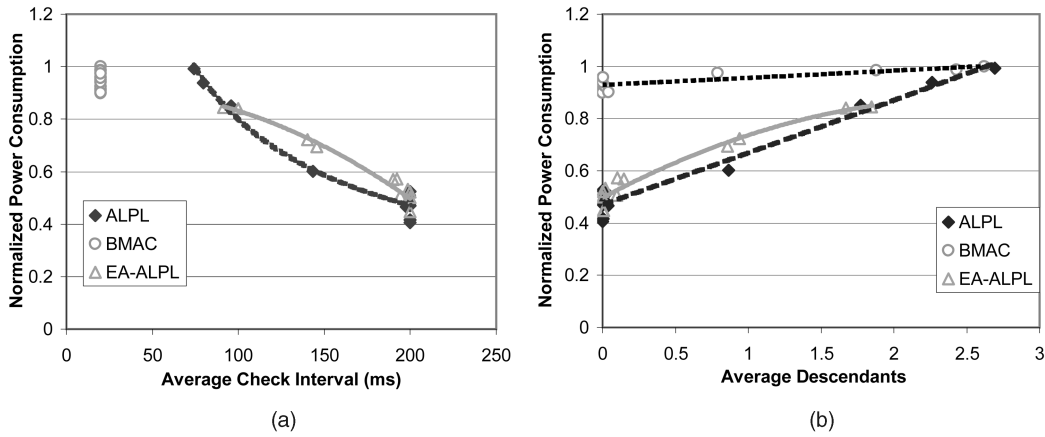


Fig. 9. (a) Local power consumption as a function of average check interval. (b) Local power consumption as a function of the average number of descendants.

number of descendants variable is the main contributor for the reductions in energy consumption, as the next subsection reveals in more detail. This reduction also confirms the benefits of greedy local decisions in reducing overall network power consumption.

The global energy consumption for ALPL and EA-ALPL experiments is almost the same with a slight difference of 1 percent. The main distinction between ALPL and EA-ALPL is in the distribution of the power consumption among network nodes rather than in the aggregated power consumption. The remainder of this subsection explores the local power consumption at individual nodes, which uncovers the load balancing benefits of EA-ALPL.

### 5.1.3 Local Power Consumption

We now turn our attention to local power consumption at each node. We collect node statistics by piggybacking node status information into data packets. Each arriving data packet for a node provides one data point regarding node status. Recall that during each 43 hour experiment, each node sends 2,580 data packets, yielding 2,580 data points for each node.

Fig. 8b shows the average check interval of each node in the BMAC, ALPL, and EA-ALPL deployment experiments, with the error bars indicating the 99 percent confidence interval. In plain BMAC, all nodes use a check interval of 20 ms. The average check interval for ALPL and EA-ALPL ranges between 20 ms and 200 ms throughout the deployment. In both ALPL and EA-ALPL, the average check intervals of all nodes are longer than in BMAC, so all nodes save on idle listening power consumption.

EA-ALPL yields a more balanced spread of check intervals among nodes than ALPL or BMAC. The nodes with the lowest average check interval in ALPL, such as nodes 1 and 8, have a higher average check interval with EA-ALPL. Similarly, some of the nodes with the highest average check intervals in ALPL have lower check intervals in EA-ALPL. The balancing out of average check intervals yields more balanced energy consumption among network nodes, as Fig. 9a reveals.

The narrow 99 percent confidence interval for all nodes in the ALPL and EA-ALPL experiments indicates the

stability of this approach in the long-term experiments. We also note that busier nodes with the smaller average check interval had slightly less stable check intervals compared to other nodes due to occasional routing oscillations. The stability of check intervals for busy nodes is of the same order for both ALPL and EA-ALPL. Since the basic version of ALPL does not introduce any routing cost modifications, we attribute these transient oscillations to link state changes in the dynamic network topology.

Fig. 9a shows the effect of average check intervals on total local power consumption at each node for the duration of each experiment. Each point in Fig. 9a relates the aggregated power consumption of a single node during an experiment to the node’s average check interval. In plain BMAC, all nodes consume almost the same power as the busiest node because all nodes use the same listening mode. In ALPL, nodes with an average check interval close to 200 ms achieve energy savings of more than 50 percent compared to the busiest node. EA-ALPL yields a more balanced traffic load, as it reduces power consumption at the most active nodes by about 16 percent at the cost of small increases in power consumption at less active nodes. This trade-off is favorable because network lifetime depends on the most active critical nodes.

Through a similar representation, Fig. 9b plots the aggregate local power consumption of each node based on the node’s average number of descendants during the deployment. For both ALPL and BMAC, the range and distribution of the number of descendants is the same because both methods use the same routing metrics. This reinforces the claim in Section 3.3.1. EA-ALPL incorporates radio activity into routing decisions to balance the forwarding load, so the most loaded node in EA-ALPL has an average of two descendants in comparison with an average of 2.5 in ALPL and BMAC.

Power consumption for the BMAC case is correlated with the number of descendants, but the variation is limited. The overall trend for both ALPL experiments is that it yields more energy savings for nodes with fewer descendants because these nodes can use longer check intervals. As the number of descendants increases, the power savings of using ALPL are reduced because the

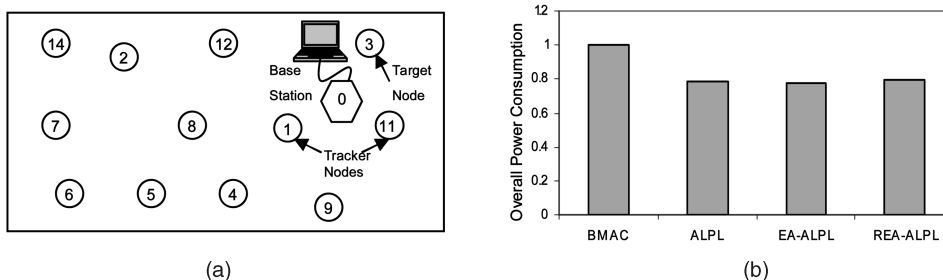


Fig. 10. (a) Physical network topology for the event-driven experiments. (b) Global network power consumption during deployment.

average check interval gets closer to the case of network-wide listening modes. EA-ALPL conforms to the trend of larger power savings for nodes with fewer descendants. However, EA-ALPL yields higher energy savings than ALPL at the nodes with the highest number of descendants, mainly by shifting the forwarding load away from busier nodes when possible. By reducing the power consumption of the busiest node, the inclusion of radio duty cycle prolongs network lifetime.

## 5.2 Event-Driven Sensor Network

In this section, we consider an event-driven sensor network application to investigate the benefits of using role information to alter network behavior. Fig. 10a shows the topology of the test-bed network. The application models a target tracking scenario where the nodes react to the appearance of the target by providing data more frequently. The nodes in the network collect and send their sensor data periodically every 60 seconds by default. We designate one of the nodes, node 3, as the target node. Nodes that detect the target node's presence, nodes 1 and 11 in Fig. 10a, begin sampling their sensors and sending the data at 30 second intervals. Nodes that do not detect the presence of the target node continue sampling their sensors at 60 second intervals.

We conduct four experiments for the same physical network topology. The first three experiments use BMAC, ALPL, and EA-ALPL as in the time-driven case above. The fourth experiment considers the node's number of descendants, duty cycle, and role as the local states and adapts network behavior accordingly. We refer to this case as Role and Energy Aware ALPL (REA-ALPL). The weights of both the radio duty cycle and sensing cost metrics,  $\alpha$  and  $\beta$ , are set to 2 for this scenario.

### 5.2.1 Global Power Consumption

The average data yield for BMAC, ALPL, EA-ALPL, and REA-ALPL remains at 98.5 percent. As in the time-driven case, data delivery is not affected by the introduction of ALPL.

Fig. 10b plots the average global network power consumption for BMAC, ALPL, EA-ALPL, and REA-ALPL. All three state representations in ALPL yield almost the same overall power consumption, since they all include the number of descendants variable, which is the main contributor to reduction of global power consumption. ALPL reduces the overall network power consumption by 21 percent over the BMAC case for this event-driven network. The reductions in global power consumption are

lower than the time-driven network case because the sensing power consumption constitutes a larger portion of overall power consumption in the event-driven case.

### 5.2.2 Local Power Consumption

We now turn our attention to the local power consumption at each node. Before presenting the local power consumption results, we note that the goal of this analysis is to illustrate how the addition of the role aspect contributes to load balancing, leading to a longer network lifetime. Ideally, all nodes would deplete their batteries at about the same time. In this experiment, the goal is to explore the extent to which nodes can recognize the increased sensing activity of the tracker nodes 1 and 11 in order to favor these nodes in routing decisions.

We first examine the average check interval for each node during the deployment. Fig. 11a plots the average check interval of each node for the duration of each of the three ALPL experiments. We omit the constant BMAC check interval from the figure for presentation clarity. In the ALPL experiment, the tracker node with an ID of 1 is the busiest node with a check interval of 43 ms. By including radio duty cycle information, EA-ALPL shifts some of node 1's forwarding load to node 9, enabling a slightly longer average check interval at node 1. REA-ALPL includes information on sensing activity, which shifts most of node 1's forwarding load to nodes 8 and 9. As a result, we observe an average check interval of 183 ms at node 1 in the REA-ALPL, while the check intervals of nodes 8 and 9 drop to 51 and 75 ms, respectively. The expanded state representation of REA-ALPL did not cause any added instability in the network, evidenced by the similar width of error bars for the three experiments.

Fig. 11b examines the topological distinctions of each experiment by plotting the average number of descendants for each node in the ALPL experiments. Fig. 11b omits the results for the BMAC experiment results because the number of descendants is the same as ALPL. For the case of ALPL, the top three forwarders have a number of descendants that ranges between 2.62 and 1.1. EA-ALPL considers radio duty cycle as the main reason for energy imbalance, which narrows the gap in the average number of descendants of the three busiest nodes to a range between 1.875 and 1.32. REA-ALPL provides a more expressive node energy profile by considering the potential for increased sensing frequency at tracker nodes. REA-ALPL recognizes that node 1 already has a higher power burden for tracking the target node and reporting the data at double the

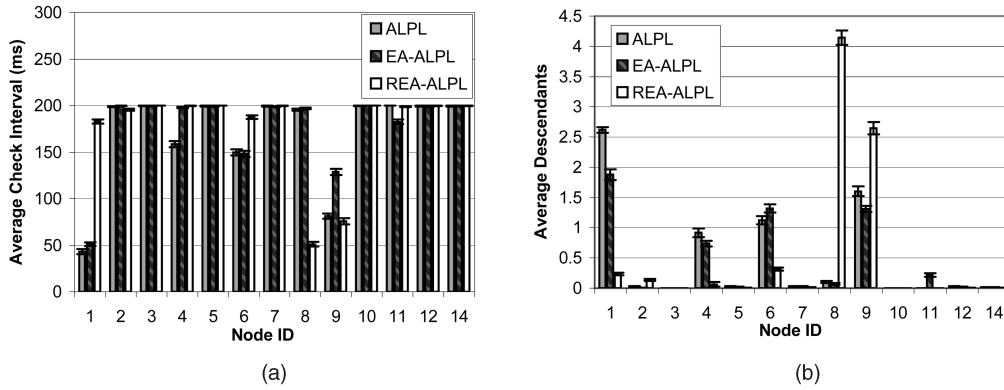


Fig. 11. (a) Average check intervals at each node. (b) Average number of descendants. The error bars show the 99 percent confidence intervals.

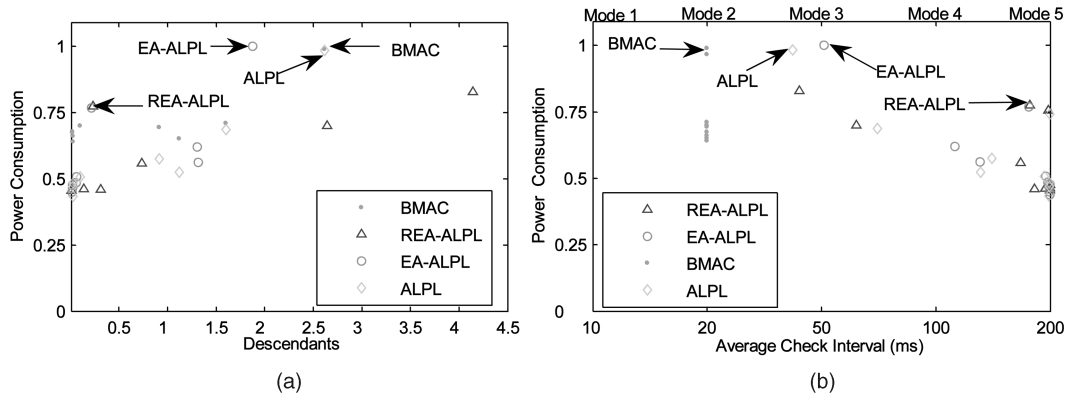


Fig. 12. (a) Power versus average number of descendants. (b) Power versus average check interval.

frequency of other nodes, so it relieves node 1 from most of its forwarding burden. As a result, REA-ALPL forces nodes to avoid node 1 as a routing parent, which shifts most of the forwarding to nodes 8 and 9. The number of descendants at nodes 8 and 9 experiences higher instability than at other nodes in the REA-ALPL experiment, but the absolute instability in the number of descendants for all nodes appears to be small and transient.

Fig. 12a shows the effect of shifting the descendants away from node 1. The labeled arrows in Fig. 12a indicate the point corresponding to node 1 for the four cases. The power consumption for node 1 in the cases of BMAC, ALPL, and EA-ALPL is about the same although EA-ALPL reduces the number of descendants by about 25 percent. This effect is explained by the dominant effect of sensing power consumption of node 1 over its listening power consumption. REA-ALPL reduces the power consumption of node 1 by about 25 percent. By shifting the descendants of node 1 to nodes 8 and 9, REA-ALPL enables node 1 to have a longer lifetime than node 8 and keep on reporting the target node even after node 8 dies. REA-ALPL thus provides the most balanced energy consumption of the four experiments through an extended cross-layer state representation that considers all the causes of power consumption imbalance.

Shifting descendants away from node 1 reduces both the reception and transmission power consumption at node 1, but it also reduces the idle listening power consumption at

node 1 as indicated by Fig. 12b. In contrast, the three experiments that do not consider node role fail in reducing node 1’s listening power consumption. Because REA-ALPL diverts almost all forwarding traffic away from node 1, it enables node 1 to have an average check interval of about 183 ms. This increase in check interval significantly lowers the listening power consumption at node 1 and reduces overall power consumption at this node. The traffic diverted from node 1 to node 8 causes node 8 to have the shortest average check interval among all three state representations. As mentioned before, the traffic shift enables power consumption balancing in the network. As a result, the plot for REA-ALPL exhibits the lowest variation in the power consumption of the four most loaded nodes for all experiments.

Through its consideration of a more comprehensive definition of node state that includes role information, REA-ALPL has yielded better load balancing in the event-driven tracking network experiments. In particular, it has shifted the traffic load away from the nodes with high sensing activity. This traffic shift has reduced the power consumption of the busiest node in the network by 23 percent, resulting in significant improvement in the longevity of the critical tracker nodes.

## 6 DISCUSSION

This paper has proposed a framework for greedy cross-layer local optimizations in sensor networks that reduces

overall power consumption in the network and promotes load balancing among nodes to customize network behavior to an application's performance requirements. The framework enables nodes to use their local and neighborhood state information to determine their behavior at the MAC layer and at the network layer. At the network layer, a flexible cost function enables nodes to customize routing cost metrics according to an application's performance requirements. At the MAC layer, the nodes set their check intervals in BMAC according to their local state. Bringing together the optimizations at the network and MAC layers, ALPL ensures seamless adaptation to local node state.

We have validated the framework through two sets of experiments on a testbed of sensor nodes. The first set of experiments represented a typical time-driven monitoring sensor network with a single data sink. The second set of experiments was an event-driven network that models a target-tracking application.

The experiments have explored three representations for local node state. The analysis and results show that including more information from across the network stack into the local state representation better reflects nodes' energy profiles and enables more informed adaptations of network behavior. In the time-driven case, the state representation combining duty cycle and descendants yields the most balanced power consumption distribution. In the event-driven case, the state representation that includes role, duty cycle, and descendants yields the most balanced spread of power consumption for the event-driven case.

The adaptive and flexible nature of our framework supports the dynamic nature of sensor networks and can exploit local state information about the present, the past, and the predicted future state of sensor nodes to reduce power consumption. We have studied how adapting listening modes to the node's current logical topology position (which represents the node's present state), duty cycle (which represents the node's past state), and role (which represents the node's predicted future state) can reduce the global network power consumption and the local power consumption at each node.

The use of a proactive routing strategy for sensor networks requires careful tuning of the routing update period for the network application in order to balance route adaptivity and energy efficiency. In long term monitoring applications, sending route update message with a period of the same order of data message periods ensures that the communication overhead of proactive routing is small.

One concern of using greedy approaches is that local decisions may not be globally optimal. For instance, a well-known case for greedy approaches is when the cheapest next hop does not represent the best path to the destination. In our framework's implementation, the shortest path algorithm in MintRoute uses a modified cost metric including link quality and power to route packets to the next hop. Because each node learns its hop count to the base station by adding a single hop to its parent hop count, nodes are guaranteed to have an accurate view of their hop count. In contrast, nodes learn the link quality and energy cost metrics for their direct neighbors only, so nodes may not adapt instantaneously to abrupt changes for these metrics elsewhere in the network. However, we have observed through our deployments that the nodes' energy

cost metrics change gradually rather than abruptly. As for link qualities, transient changes occur due to movement of objects around the nodes, but the high data yield rate in our deployment has shown that these changes have a minimal effect on data delivery or energy savings.

Our framework adopts BMAC for the MAC layer protocols and introduces ALPL to interface the MAC layer with a proactive strategy at the routing layer. For any proactive routing protocol, ALPL does not significantly increase communication, processing, or storage complexity. The results in Section 5 have shown that the communication overhead of a few bits of state information in periodic routing messages is a small price to pay for the energy savings of ALPL. In terms of processing complexity, cost metric calculations in ALPL involve simple arithmetic operations and routing decisions involve a few simple conditional statements. Finally, the state information stored at each node is a function of node density and not network size, which adds scalability to the greedy approach.

Because of the reduced transmission power in our deployment and the limited space in the laboratory, each node had an average of about four neighbors, yielding a relatively high network density. The high node density in our experiments raised the degree of contention among nodes. Coupled with the framework's locally optimal decisions, the experiments have confirmed that the framework is scalable to large or dense networks despite the limited size of our network test-bed.

Reduction of global network power consumption through local decisions is an approach that is widely applicable to many sensor network applications and quality of service requirements. Our framework is independent of the underlying routing protocol or MAC protocol. Instead, it can build on other underlying mechanisms with a modular design to optimize the behavior of any sensor network application.

## ACKNOWLEDGMENTS

This work was partially supported by US National Science Foundation grant No. CCF-0347902. The work of Raja Jurdak has been partially supported by IRCSET.

## REFERENCES

- [1] B. Karp and H.T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *Proc. MobiCom*, 2000.
- [2] P. Skraba, H. Aghajan, and A. Bahai, "Cross-Layer Optimization for High Density Sensor Networks: Distributed Passive Routing Decisions," *Proc. Int'l Conf. Ad-Hoc, Mobile, and Wireless Networks (ADHOC-NOW)*, pp. 266-279, 2004.
- [3] R.L. Cruz and A.V. Santhanam, "Optimal Routing, Link Scheduling and Power Control in Multi-Hop Wireless Networks," *Proc. INFOCOM*, 2003.
- [4] T. ElBatt and A. Ephremides, "Joint Scheduling and Power Control for Wireless Ad Hoc Networks," *IEEE Trans. Wireless Comm.*, vol. 1, pp. 74-85, 2004.
- [5] U.C. Kozat, I. Koutsopoulos, and L. Tassiulas, "A Framework for Cross-Layer Design of Energy-Efficient Communication with QoS Provisioning in Multi-Hop Wireless Networks," *Proc. INFOCOM*, 2004.
- [6] R. Jurdak, "Modeling and Optimization of Ad Hoc and Sensor Networks," PhD dissertation, Bren School of Information and Computer Science, Univ. of California Irvine, Sept. 2005.
- [7] P. Baldi, L. De Nardis, and M.G. Di Benedetto, "Modelling and Optimization of UWB Communication Networks through a

- Flexible Cost Function," *IEEE J. Selected Areas in Comm.*, vol. 20, no. 9, pp. 1733-1744, 2002.
- [8] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," *Proc. ACM Conf. Embedded Networked Sensor Systems (SenSys '04)*, 2004.
- [9] R. Jurdak, P. Baldi, and C.V. Lopes, "Energy-Aware Adaptive Low Power Listening for Sensor Networks," *Proc. Int'l Workshop Networked Sensing Systems (INSS '05)*, 2005.
- [10] R. Jurdak, P. Baldi, and C.V. Lopes, "State-Driven Energy Optimization in Sensor Networks," *Proc. IEEE Int'l Conf. Sensor Networks (SENET '05)*, 2005.
- [11] A. Woo and D. Culler, "A Transmission Control Scheme for Media Access in Sensor Networks," *Proc. MobiCom*, 2001.
- [12] D. Culler et al., "Towards a Sensor Network Architecture: Lowering the Waistline," *Proc. Workshop Hot Topics in Operating Systems (HotOS X)*, 2005.
- [13] J. Polastre et al., "A Unifying Link Abstraction for Wireless Sensor Networks," *Proc. ACM Conf. Embedded Networked Sensor Systems (SenSys '05)*, Nov. 2005.
- [14] K. Romer and F. Mattern, "Event-Based Systems for Detecting Real-World States with Sensor Networks: A Critical Analysis," *Proc. DEST Workshop at Int'l Conf. Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2004.
- [15] J. Liu et al., "State-Centric Programming for Sensor and Actuator Network Systems," *IEEE Pervasive Computing Magazine*, vol. 2, no. 4, 2003.
- [16] V. Mhatre et al., "A Minimum Cost Heterogeneous Sensor Network with a Lifetime Constraint," *IEEE Trans. Mobile Computing*, vol. 4, no. 1, 2005.
- [17] W. Ye and J. Heidemann, "Medium Access Control in Wireless Sensor Networks," *Wireless Sensor Networks*, T. Znati, K.M. Sivalingam, and C. Raghavendra, eds., Kluwer Academic Publishers, 2003.
- [18] *IEEE 802.15.4, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, IEEE, <http://standards.ieee.org/getieee802/download/802.15.4-2003.pdf>, 2003.
- [19] R. Jurdak, C.V. Lopes, and P. Baldi, "A Survey, Classification, and Comparative Analysis of Medium Access Control Protocols for Ad Hoc Networks," *IEEE Comm. Surveys and Tutorials*, vol. 6, no. 1, 2004.
- [20] A. El-Hoiydi, J.D. Decotignie, and J. Hernandez, "Low Power MAC Protocols for Infrastructure Wireless Sensor Networks," *Proc. Fifth European Wireless Conf.*, Feb. 2004.
- [21] Q. Jiang and D. Manivannan, "Routing Protocols for Sensor Networks," *Proc. Consumer Comm. and Networking Conf. (CCNC '04)*, 2004.
- [22] C.E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *Proc. SIGCOMM*, pp. 234-244, 1994.
- [23] T. Chen and M. Gerla, "Global State Routing: A New Routing Scheme for Ad-Hoc Wireless Networks," *Proc. Int'l Conf. Comm. (ICC '98)*, 1998.
- [24] C.E. Perkins, E.M. Royer, and S.R. Das, "Ad Hoc On-Demand Distance Vector Routing," IETF Internet draft, work in progress, Oct. 1999.
- [25] D.B. Johnson and D.A. Maltz, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks," IETF Internet draft, work in progress, Oct. 1999.
- [26] C.K. Toh, "Maximum Battery Life Routing to Support Ubiquitous Mobile Computing in Wireless Ad Hoc Networks," *IEEE Comm. Magazine*, pp. 138-147, June 2001.
- [27] R.C. Shah and J. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks," *Proc. Wireless Comm., and Networking Conf. (WCNC '02)*, 2002.
- [28] C. Park, K. Lahiri, and A. Raghunathan, "Battery Discharge Characteristics of Wireless Sensor Nodes: An Experimental Analysis," *Proc. IEEE Comm. Soc. Conf. Sensor and Ad Hoc Comm. and Networks (SECON '05)*, 2005.
- [29] R. Szewczyk et al., "Habitat Monitoring with Sensor Networks," *Comm. ACM*, vol. 47, no. 6, pp. 34-40, 2004.
- [30] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. INFOCOM*, 2002.
- [31] T. Van Dam and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. Int'l Conf. Embedded Networked Sensor Systems*, pp. 171-180, 2003.
- [32] W. Ye, J. Heidemann, and D. Estrin, "Medium Access Control with Coordinated, Adaptive Sleeping for Wireless Sensor Networks," *IEEE Trans. Networking*, Apr. 2004.
- [33] M. Haenggi, "Energy-Balancing Strategies for Wireless Sensor Networks," *Proc. IEEE Int'l Symp. Circuits and Systems (ISCAS '03)*, 2003.
- [34] R. Szewczyk et al., "Lessons from a Sensor Network Expedition," *Proc. European Conf. Wireless Sensor Networks (EWSN '04)*, 2004.
- [35] "Tiny Operating System," <http://tinynos.net>, 2005.
- [36] A. Woo, T. Tong, and D. Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks," *Proc. ACM Conf. Embedded Networked Sensor Systems (Sensys '03)*, 2003.
- [37] R. Sivakumar, B. Das, and V. Bharghavan, "The Clade Vertebrata: Spines and Routing in Ad Hoc Networks," *Proc. IEEE Symp. Computers and Comm. (ISCC '98)*, 1998.
- [38] R. Jurdak, C.V. Lopes, and P. Baldi, "Battery Lifetime Estimation and Optimization for Underwater Sensor Networks," *Sensor Network Operations*, May 2006.
- [39] Crossbow Technology Inc., "Mica 2 Motes," <http://www.xbow.com/Products/productdetails.aspx?sid=72>, 2005.
- [40] D. Gay et al., "The nesC Language: A Holistic Approach to Networked Embedded Systems," *ACM SIGPLAN Notices*, vol. 38, no. 5, pp. 1-11, 2003.



**Raja Jurdak** received the PhD degree in information and computer sciences at the University of California, Irvine in 2005. From 2005 to 2006, he was a postdoctoral researcher at the University of California, Irvine. He is currently a senior researcher in the Adaptive Information Cluster at University College Dublin. He was a recipient of the IRCSET Embark fellowship in 2006. Dr. Jurdak is the author of more than 21 refereed journal and conference publications, as well as the book *Wireless Ad Hoc and Sensor Networks: A Cross-Layer Design Perspective* (Springer, 2007). His research focuses on application-driven networks, modeling of ad hoc and sensor networks, emerging communication technologies, underwater acoustic networks, and cross-layer design. He is a member of the IEEE and the IEEE Computer Society.



**Pierre Baldi** received the PhD degree in mathematics from the California Institute of Technology in 1986. From 1986 to 1988, he was a postdoctoral fellow at the University of California, San Diego. From 1988 to 1995, he was a member of the faculty and technical staff at the California Institute of Technology and at the Jet Propulsion Laboratory (JPL). He was CEO of a startup company from 1995 to 1999 and joined the University of California, Irvine (UCI) in 1999. He is currently a chancellor professor in the School of Information and Computer Sciences, director of the Institute for Genomics and Bioinformatics, and member of the California Institute for Telecommunications and Information Technology (Calit2) at UCI. He is the recipient of a 1993 Lew Allen Award at JPL and a Laurel Wilkening Faculty Innovation Award at UCI. Dr. Baldi is the author of more than 200 research articles and four books, including *Modeling the Internet and the Web—Probabilistic Methods and Algorithms* (Wiley, 2003). His research focuses on probabilistic modeling and statistical inference, machine learning, bioinformatics, data mining, and communication networks. He is a senior member of the IEEE.



**Cristina Videira Lopes** is an associate professor at the University of California, Irvine. She conducts research in programming languages, software engineering, and pervasive computing. She is the recipient of a US National Science Foundation CAREER Award. She is a member of the IEEE.