

State-Driven Energy Optimization in Wireless Sensor Networks

Raja Jurdak
UCI/School of ICS
California Inst. of Telecomm.
and Information Tech. Calit2
rjurdak@ics.uci.edu

Pierre Baldi
UCI/School of ICS
California Inst. of Telecomm.
and Information Tech. Calit2
pfbaldi@ics.uci.edu

Cristina Videira Lopes
UCI/School of ICS
California Inst. of Telecomm.
and Information Tech. Calit2
lopes@ics.uci.edu

Abstract

Most sensor network applications require quality of service guarantees on a network-wide basis, suggesting the need for global network cost optimization. The dynamic and nonuniform local states of individual nodes in sensor networks complicate global cost optimization. Here, we present an approach for optimizing global cost in sensor networks through greedy local decisions at each node, and we explore the benefits of this approach in reducing the idle listening at individual nodes in order to reduce the global network energy cost. We consider two representations for the local sensor node state: (1) number of descendants in the routing tree; and (2) number of descendants and duty cycle. For both state representations, we show through experiments on a testbed of 14 mica2 sensor nodes running ALPL that enabling nodes to set their listening mode according to their local state reduces global energy cost by 35% and provides more balanced energy consumption over the case of BMAC.

1 Introduction

Advances in processor, memory, communication and sensing technology have fueled increased interest in sensor networks and their wide range of potential applications. Within the application space for sensor networks, each applications' objectives may require a custom set of quality targets. In many cases, the goal is to meet the quality targets for the network as a whole rather than on a per-node basis, which suggests the need for global network cost optimization. Global network cost optimization is particularly challenging for wireless sensor networks because the state of individual sensor nodes is highly dynamic and nonuniform.

To address these challenges, we adopt a collaboration strategy that enables each node to greedily optimize its local cost based on its local and neighborhood state information. Nodes can easily learn their own state and their neighbors' states by piggybacking state information in periodic routing messages, which minimizes overhead communication. The low additional communication of this strategy makes it the

most suitable for sensor networks. The optimal local decisions yield significant reductions in overall network cost with minimal communication overhead.

In this paper, we explore the benefits of optimizing global network energy cost through greedy local decisions made by each node based on its local and neighborhood state. Enabling individual nodes to adjust their behavior according to their local state requires underlying mechanisms that are adaptive, flexible and modular.

In order to investigate the validity of our approach, we choose BMAC [1], 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 using these interfaces to set listening and transmit modes according to additional information on the application and operation of a sensor network could produce further power savings for BMAC.

Building on BMAC, our work proposes 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. In original BMAC, setting a network-wide listening mode disregards the non-uniform and dynamic local states of individual nodes. Per node listening modes are 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 study considers two different representations for local node states in a single data sink network: (1) number of descendants in the routing tree rooted at the base station; and (2) number of descendants and duty cycle. A node's number of descendants indicates the node's forwarding load. For instance, leaf nodes have no forwarding load. Our second representation of state incorporates knowledge about the duty cycles of neighbors into the decision of optimal listening mode. The duty cycle quantifies how busy a node has been.

We show through experiments on a testbed of sensor

nodes that for both state representations, enabling each node to select its optimal listening mode according to its local state reduces the global network cost. We also compare the resulting global network cost for the two state representations, and we investigate the energy benefits of adding knowledge about duty cycle, especially in the promotion of load balancing in the network.

The rest of the paper is organized as follows. Section 2 discusses related work, including BMAC. Section 3 provides an overview of ALPL. Section 4 provides the analytical justification for reducing global cost through local decisions in ALPL. Section 5 presents our experiments to validate the approach on mica2 motes. Section 6 discusses the results and concludes the paper.

2 Related Work

Previous efforts have realized the need for optimizing global cost in wireless networks. Baldi et al. [2] develop a global cost function for wireless Ultra Wide Band 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. [3] optimize the hardware cost of heterogeneous sensor networks with a lifetime constraint. Their work considers a network with 2 types of nodes, and attempts to determine both the number and position of each node to minimize overall hardware cost while satisfying the lifetime constraint.

In sensor networks, energy efficiency is the primary cost metric of interest. Many protocols have been designed to provide energy-efficient behavior at both the MAC layer [4, 5] and the routing layer [6]. 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 [7]. Thus, energy-efficient MAC protocol proposals have focused on minimizing idle listening at sensor nodes [1, 8].

The recent work by Pollastre et al. proposes BMAC [1]. 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 8 check intervals, and each check interval corresponds to one of BMAC's 8 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 8 different preamble lengths referred to as transmit modes. Additionally, Pollastre 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.

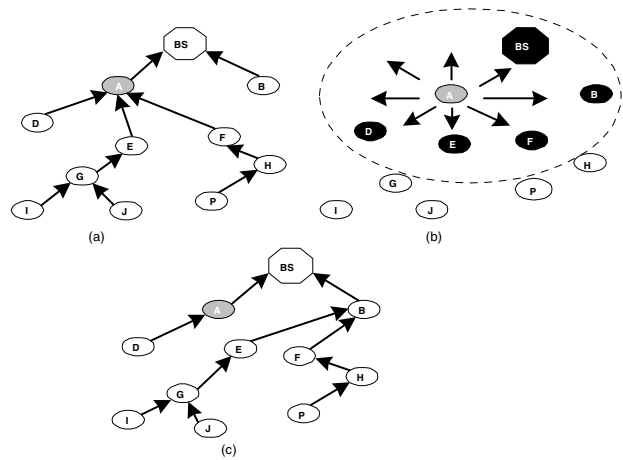


Figure 1. Node Interaction in ALPL

3 Adaptive Low Power Listening

Adaptive Low Power Listening [9] is a cross-layer mechanism that builds on BMAC and that adapts to dynamic sensor network topologies and nonuniform energy consumption. It enables nodes to locally optimize their listening modes while ensuring correct data delivery. This work explores the impact of setting listening modes according to a node's number of descendants in the routing tree and a node's duty cycle.

In this section, we first describe the main steps involved in ALPL that enable nodes to dynamically set their listening mode and to adaptively set their transmit mode. Then, we discuss the main design choices for ALPL. Finally, we present the routing modifications required for ALPL.

3.1 Description

ALPL enables a node to set its own listening mode according to its current state, and to adapt its transmit mode to fit the listening mode of its routing parent. Figure 1 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 state 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 towards the base station (Figure 1(a)). As a result, each node learns its local state and 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 lis-

tening mode and state information along with other routing information (Figure 1(b)).

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.

3.2 Node-specific Listening Modes

In general, individual node states in sensor networks are not predictable nor static [12]. Node states are affected by factors such as interference variations and dynamic node membership. Thus, network designers have to make conservative assumptions in determining network configuration. In the case of BMAC, conservative assumptions lead to setting network-wide listening and transmit modes 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 state. The rationale is that in dynamic sensor networks, each node always has the most up-to-date view of its own local state [13]. Node states can be defined by the network designer or operator depending on the applications' goals and quality of service requirements.

3.3 Number of Descendants

Our first representation of state considers a node's number of descendants in the routing tree. ALPL reduces idle listening at each node by enabling the node to select the optimal listening mode for its current number of descendants in the routing tree. Optimal per-node listening modes minimize the local energy consumption at each node and contribute to reducing the global network energy consumption.

We contend that for monitoring applications, using the number of descendants to choose the optimal listening modes is more energy-efficient than using the number of neighbors, as suggested in [1]. In monitoring applications, data flow is typically towards 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. Setting the listening mode according to the neighborhood size is a more conservative option, which is partially related to the requirements of the development platform and the routing protocol. Simple modifications to the

routing protocol enable nodes to use ALPL without sacrificing data yield. The routing modifications to accommodate ALPL are described further in section 3.5.

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 γ that it forwards during a route update interval. Thus, learning the number of descendants involves no overhead communication. The number γ 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 γ packets. Then, N sends a routing update message that includes its new listening mode along with other routing information.

3.4 Duty Cycle

Our second state representation combines the number of descendants and the node's duty cycle. The dependence of the listening mode on duty cycle addresses the inherently non-uniform energy consumption in sensor networks [3, 10, 11]. Highly loaded nodes deplete their battery resources at a faster rate than other nodes in the network. ALPL can help balance the energy cost in the network by enabling individual nodes to adapt their listening mode according to their power consumption rate. While the number of descendants at a node reveals the current activity of a node, the node's duty cycle indicates the node's past activity, in particular, the node's energy consumption in the previous time window. A high duty cycle indicates that the node has been highly active up till the present point in time and vice versa.

Sharing duty cycle information among node 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 power states of all their one-hop neighbors and store this information in their local neighbor table.

The main idea of load balancing in ALPL is to shift energy cost among network nodes to achieve more uniform energy consumption. More specifically, ALPL can divert some multihop traffic from nodes with high duty cycle to less active nodes in order to allow highly active nodes to listen less often and to balance the load. ALPL implicitly uses a node's duty cycle in setting the node's listening mode. A high duty cycle at a particular node causes its neighbors to increase its routing cost (see section 3.5).

As an example, consider Figure 1 again. The routing tree in Figure 1(a) 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 (Figure 1(c)). 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.

3.5 Routing Modifications

Our primary platform for sensor network development is TinyOS [14], developed at UC Berkeley. Within TinyOS, the standard routing protocol is called MintRoute. MintRoute is a proactive routing protocol in which nodes send periodic routing messages to declare their local states. The original cost metric in MintRoute combines hop count and link quality.

The main purpose of both the hop count metric and the link quality metric is to maximize data delivery by choosing the shortest and most reliable path to the base station. Both hop count and link quality also implicitly enforce energy efficiency. Hop count reduces the number of nodes involved in forwarding activity. Link quality reduces the number of retransmissions by favoring more reliable links. Since balancing energy cost in the network is one of our main goals, we introduce an explicit energy cost metric that depends on duty cycle.

The rest of this section discusses routing modifications to ensure that ALPL is compatible with the routing protocol for the purposes of: (1) data delivery; and (2) energy awareness.

3.5.1 Data Delivery

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, MintRoute defines quality as the percentage of data packets correctly received from a neighbor. The example above on missed data packets causes the quality metric in MintRoute to drop. Consequently, 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.

3.5.2 Energy Awareness

In all proactive routing protocols, each node periodically selects its routing parent. In MintRoute, 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:

$$C(M) + \epsilon < C(RP) \quad (1)$$

where $C(N_j)$ is the routing cost of node N_j , and ϵ is the switching threshold that ensures that a node switches its routing parent only when there is an appreciable benefit in doing so.

Our extended local state representation considers a node's energy state as well as its number of descendants. In order to integrate energy-awareness into MintRoute, the choice of a routing parent must also depend on how busy a node has been relative to its neighbors. Highly loaded nodes should have a higher routing cost. We introduce a new cost metric $C(power)$ for a neighbor N_j :

$$C(power) = \epsilon \frac{\delta_j - \sum_{i=0}^k \delta_i/k}{\sqrt{\sum_{i=0}^k \delta_i^2/k - (\sum_{i=0}^k \delta_i/k)^2}} \quad (2)$$

where k is the number of neighbors in the node's local neighbor table, and δ_i is the duty cycle of neighbor N_i . Equation 2 compares the duty cycle of neighbor N_j to the average duty cycle in the neighborhood and normalizes the difference. The normalized difference determines the extent of statistical deviation of the power state of N_j among all the nodes in the neighborhood.

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, $C(power)$ 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 cost of quality and hops, as well as the power cost according to the following equation:

$$C_{new}(M) = \begin{cases} C(M) + \alpha C(power) & lev(M) \leq lev(RP) \\ C(M) & lev(M) > lev(RP) \end{cases} \quad (3)$$

where α is a constant representing the weight of $C(power)$, and lev represents the number of hops of a node from the base station. With the new cost definition, neighbors with a higher duty cycle have a higher routing cost, and they are less likely to be chosen as forwarders.

4 Analysis

This section presents the analytical basis for using greedy local decisions in ALPL to reduce global network cost. 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 [1]:

$$E = E_t + E_r + E_d + E_{listen} + E_{sleep} \quad (4)$$

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 [1]. We limit the discussion here to the qualitative aspects that are relevant to ALPL.

In monitoring 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

Our ultimate goal is to minimize the local energy cost E at every node by enabling each node to locally select its own optimal listening mode while maintaining correct and timely data delivery. Minimizing E on a per-node basis reduces the total network cost and builds on the following observations about equation 4:

1. E_d and E_{sleep} are not significant factors in determining optimal listening mode. E_d is equal for all nodes throughout the network. E_{sleep} is at least an order of magnitude smaller than the other terms in equation 4, 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 N determines E_{listen} . It also determines the preamble length for packets that are received at N . Consequently, the listening mode at a node 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} . 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 .

4.2 Power State

The energy consumption in sensor networks is nonuniform among the nodes. 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 [3]. Critical nodes have a larger forwarding burden and consume more energy than nodes further away from the sink [11]. 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.

In terms of equation 4, 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. The loaded critical node informs its children to choose a new routing parent (by increasing its routing cost), thereby reducing its E_r and E_t . 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} cause the overall power consumption E at the critical node to drop.

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 [15]. Our implementation of ALPL is in NesC [16], 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 cost; and (2) to evaluate the energy savings for state-driven optimizations at individual nodes.

We conduct 3 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 nodes runs ALPL and set its listening mode according to its number of descendants in the

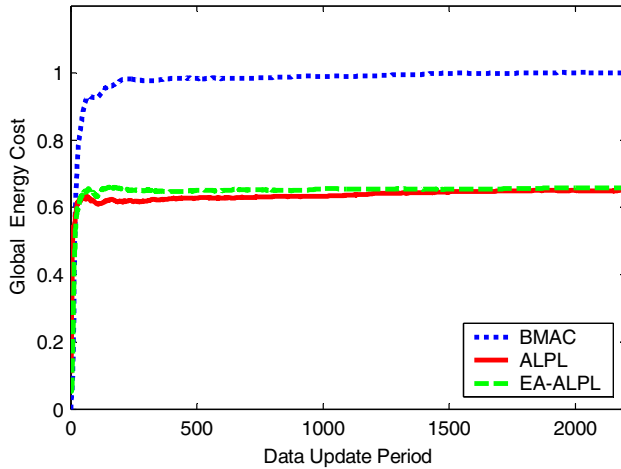


Figure 2. Global network energy cost

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. For both ALPL and EA-ALPL experiments, the routing update period is 90 seconds. The routing update period for the network-wide listening mode experiment is 120 seconds.

5.1 Global Energy Cost

The average data yield for BMAC, ALPL, and EA-ALPL is about 98.5%. Because the data yield is the same with or without state-driven optimizations, the data delivery aspect of the routing function, which is embodied in the hop count and link quality, is unaffected by state-driven optimizations (see equation 3). As a result, the differentiating cost component in the network is the energy cost.

Figure 2 plots the average global network energy cost as a function of time (denoted by data update periods) for BMAC, ALPL, and EA-ALPL. Both ALPL and EA-ALPL reduce global energy cost by about 35% on average during the deployment. The reduction in global cost stems from the optimal local decisions at each node. This reduction also confirms the benefits of greedy local decisions in reducing global network cost.

The average global cost for the 2 experiments is almost the same with a slight difference of 1%. The main distinction between ALPL and EA-ALPL is in the distribution of the global cost in the network rather than in the absolute global cost. The next subsection sheds more light on the individual node costs that drive the global cost reductions of ALPL and EA-ALPL and the details behind the local cost shifts in EA-ALPL.

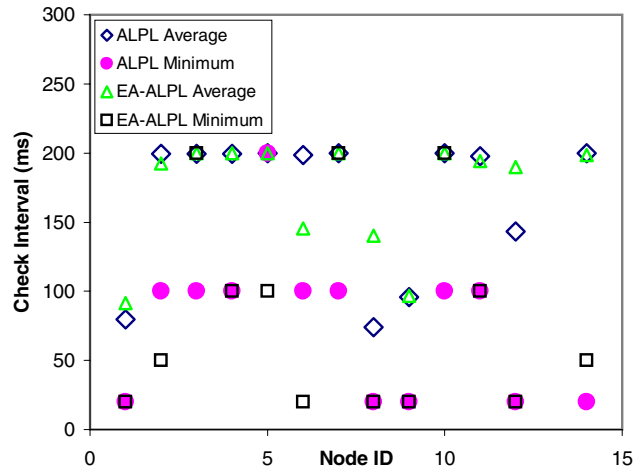


Figure 3. Average and Minimum Check intervals at each node

5.2 Local Energy Savings

Figure 3 shows the average and minimum check interval of the nodes using ALPL and EA-ALPL, and it excludes the case BMAC since the listening mode for all nodes is the same in BMAC. In Figure 3, the check interval ranges between 20 ms and 200 ms throughout the deployment. For ALPL, the average check interval for 9 of the nodes is about 200 ms, while the other 4 nodes¹ have average check intervals ranging between 74 and 143 ms. These 4 nodes are the same ones whose minimum check interval is 20 ms. The wide range of check intervals is indicative of the dynamic nature of the topology. Changes in interference conditions cause nodes to select new parents, which in turn causes the old parents to increase their check intervals.

Figure 3 also illustrates that EA-ALPL yields a more balanced spread of check intervals among nodes than ALPL or BMAC. The nodes with the lowest average check 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 reflects directly on overall power consumption. In plain BMAC, all nodes would have to use a check interval of 20 ms, so most of them would waste considerable energy on idle listening.

Figure 4 plots the local power consumption at each node as a function of its average check interval. In plain BMAC, all nodes consume almost the same power as the busiest node because all nodes use the same check interval of 20 ms. In ALPL, nodes with an average check interval close to 200 ms achieve energy savings of more than 50% compared to busiest node. EA-ALPL prolongs the average check in-

¹The 14th node is the base station with id 0. It is not included in Figure 3 because the base station never goes to sleep

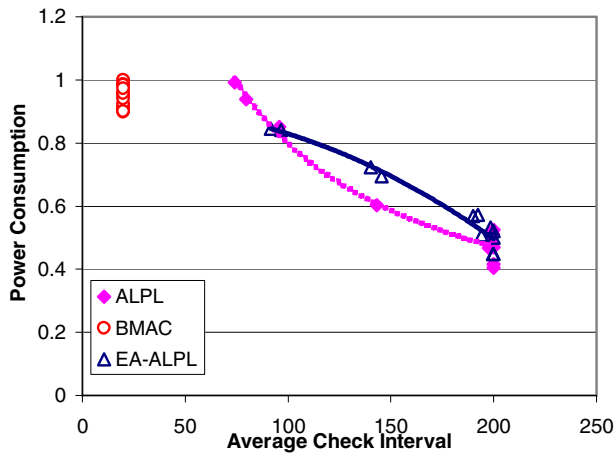


Figure 4. Power consumption vs check interval

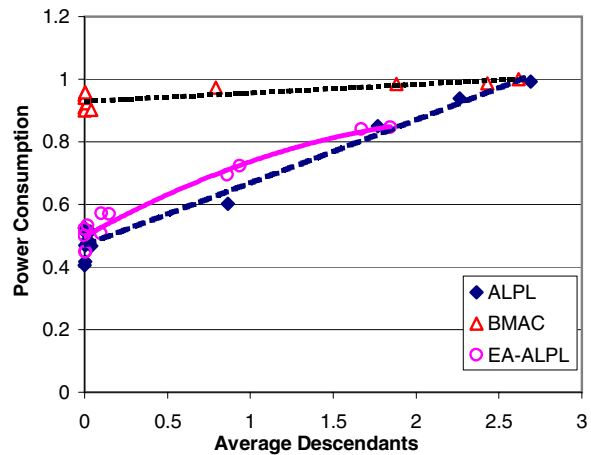


Figure 6. Power consumption vs descendants



Figure 5. Power consumption vs node level

interval at the most active nodes to about 90ms, and it reduces power consumption at these nodes by about 16% at the cost of small increases in power consumption at less active nodes. This tradeoff is favorable because network lifetime depends on the most active critical nodes.

Figure 5 compares the local power consumption according to node level. In the ALPL plot, level 2 nodes achieve more than 50% savings over BMAC. The extent of power savings at level 1 nodes varies. For highly loaded level 1 nodes, the power consumption is slightly less for ALPL than the case of BMAC. For level 1 nodes with fewer or no descendants, power savings of ALPL over BMAC are between 18 and 40%. EA-ALPL further reduces power consumption at level 1 nodes through load balancing. The tradeoff in EA-ALPL is that level 2 nodes exhibit a slightly higher power consumption than in ALPL. Since each level 1 node has fewer descendants than in ALPL, level 1 nodes choose longer check intervals. Longer check intervals at level 1

nodes cause level 2 nodes to use longer preambles, thereby increasing E_t at level 2 nodes. Thus, EA-ALPL shifts energy cost among network nodes to balance the load while achieving the same global network cost as ALPL.

Figure 6 plots the local power consumption based on the average number of descendants throughout 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. EA-ALPL incorporates energy cost into routing decisions to balance the forwarding load, so the most loaded node in EA-ALPL has an average of 2 descendants in comparison with an average of 2.5 in ALPL and BMAC.

The overall trend for ALPL and EA-ALPL is that they yield more energy savings for nodes with fewer descendants, because these nodes can use longer check intervals. Nodes with a higher number of descendants must use shorter check intervals, so they achieve fewer power savings. Nevertheless, the power consumption at the most loaded node in EA-ALPL is 16% less than the most loaded node in either ALPL or BMAC.

In short, the results of the experiments in this section confirm that considering local state asymmetries among nodes yield global energy reductions of about 35% for both ALPL and EA-ALPL. The nodes with a smaller forwarding load achieve the most significant power savings, whereas busier nodes have smaller power savings since they typically have to listen more often. The main advantage of EA-ALPL is that it shifts part of the forwarding burden from the most loaded nodes to other nodes. The degree of power savings in both mechanisms depends on the topology, interference conditions, and available redundancy in a particular network.

6 Discussion

We have proposed an approach for optimizing global cost in sensor networks through greedy local decisions at each node. We have validated the approach through experiments on a testbed of sensor nodes running ALPL, which allows nodes to adapt their listening modes in BMAC to their local state. Our proposed approach is not specific to ALPL or BMAC. Instead, it can build on any underlying mechanisms that provide flexible and modular interfaces.

In this paper, we have presented two flavors of ALPL, one that optimizes listening modes on the basis of a node's descendants in the routing graph and another that optimizes listening modes on basis of descendants and duty cycle. While both flavors provide flexible and power efficient behavior in monitoring sensor networks, EA-ALPL exhibits better behavior since it spreads energy cost more evenly in the network.

The adaptive nature of ALPL 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) and duty cycle (which represents the node's past state) can reduce the global network energy cost and the local power consumption at each node. The local state can also reflect the expected future behavior of the node, such as the node's dynamic role in the network application.

Global network cost reduction through local decisions is an approach that is widely applicable to many sensor network applications and quality of service requirements. For example, a sensor network deployed in a disaster relief area to detect signs of human activity prioritizes minimal delay and maximal reliability rather than energy cost. A global cost function for sensor networks, similar to the function in [2], can provide the flexibility required in customizing routing strategies for network applications.

References

- [1] J. Pollastre, J. Hill, and D. Culler. Versatile Low Power Media Access for Wireless Sensor Networks. Proceedings of *ACM SenSys*, 2004.
- [2] P. Baldi, L. De Nardis, and M. G. Di Benedetto. Modelling and Optimization of UWB Communication Networks Through a Flexible Cost Function. *IEEE J. Select. Areas Commun.*; 20(9): 1733–1744, 2002.
- [3] V. Mhatre, C. Rosenberg, D. Kofman et al. A Minimum Cost Heterogeneous Sensor Network with a Lifetime Constraint. *IEEE Transaction on Mobile Computing*, January 2004.
- [4] R. Jurdak, C. V. Lopes, and P. Baldi. A Survey, Classification, and Comparative Analysis of Medium Access Control Protocols for Ad Hoc Networks *IEEE Communications Surveys and Tutorials*, 6:(1), 2004.
- [5] W. Ye and J. Heidemann. Medium Access Control in Wireless Sensor Networks. *Wireless Sensor Networks*, Taieb Znati, Krishna M. Sivalingam and Cauligi Raghavendra (eds.), Kluwer Academic Publishers. 2003.
- [6] Q. Jiang and D. Manivannan. Routing Protocols for Sensor Networks. In Proc. *IEEE (CCNC)*, 2004.
- [7] R. Szewczyk et al. Habitat Monitoring with Sensor Networks. *Comm. of the ACM*, 47(6):34-40, 2004.
- [8] W. Ye, J. Heidemann and D. Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In Proc. *IEEE Infocom*, 2002.
- [9] R. Jurdak, P. Baldi, C. V. Lopes. Energy-Aware Adaptive Low Power Listening for Sensor Networks. To appear in proc. *2nd International Workshop for Networked Sensing Systems (INSS 05)*, San Diego, CA. June 2005.
- [10] M. Haenggi. Energy-Balancing Strategies for Wireless Sensor Networks *IEEE International Symposium on Circuits and Systems (ISCAS'03)*, May 2003.
- [11] R. Jurdak, C. V. Lopes and P. Baldi. Battery Lifetime Estimation and Optimization for Underwater Sensor Networks. *Sensor Network Operations*, IEEE Press, 2005 (in press).
- [12] F. Zhao and L. Guibas. *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufman Publishers ISBN 1-55860-914-8, 2004.
- [13] R. Szewczyk, J. Pollastre, A. Mainwaring, and D. Culler. Lessons from a Sensor Network Expedition. In Proc *European Workshop on Sensor Networks*, 2004.
- [14] Tiny Operating System. UC Berkeley. available: <http://tinysos.net>.
- [15] Mica 2 Motes. Crossbow Technology Inc. available: <http://www.xbow.com/Products/>
- [16] D. Gay, P. Levis, R. von Behren et al. The nesC language: A holistic approach to networked embedded systems. *ACM SIGPLAN Notices*. 38(5):1–11, 2003