Title
ASCENT: Adaptive self-configuring sEnsor networks topologies

Journal
IEEE Transactions on Mobile Computing, 3(3)

ISSN
1536-1233

Authors
Cerpa, Alberto E
Estrin, Deborah

Publication Date
2004-07-01

Peer reviewed

ASCENT: Adaptive Self-Configuring sEnsor Networks Topologies

Alberto Cerpa, *Student Member*, *IEEE*, and Deborah Estrin, *Fellow*, *IEEE*

Abstract—Advances in microsensor and radio technology will enable small but smart sensors to be deployed for a wide range of environmental monitoring applications. The low per-node cost will allow these wireless networks of sensors and actuators to be densely distributed. The nodes in these dense networks will coordinate to perform the distributed sensing and actuation tasks. Moreover, as described in this paper, the nodes can also coordinate to exploit the redundancy provided by high density so as to extend overall system lifetime. The large number of nodes deployed in these systems will preclude manual configuration, and the environmental dynamics will preclude design-time preconfiguration. Therefore, nodes will have to self-configure to establish a topology that provides communication under stringent energy constraints. ASCENT builds on the notion that, as density increases, only a subset of the nodes are necessary to establish a routing forwarding backbone. In ASCENT, each node assesses its connectivity and adapts its participation in the multihop network topology based on the measured operating region. This paper motivates and describes the ASCENT algorithm and presents analysis, simulation, and experimental measurements. We show that the system achieves linear increase in energy savings as a function of the density and the convergence time required in case of node failures while still providing adequate connectivity.

Index Terms—Wireless sensor networks, adaptive topology, topology control, energy conservation.

1 Introduction

The availability of microsensors and low-power wireless communications will enable the deployment of densely distributed sensor/actuator networks for a wide range of environmental monitoring applications from urban to wilderness environments; indoors and outdoors; and encompassing a variety of data types including acoustic, image, and various chemical and physical properties. The sensor nodes will perform significant signal processing, computation, and network self-configuration to achieve scalable, robust, and long-lived networks [2], [8], [7]. More specifically, sensor nodes will do local processing to reduce communications and, consequently, energy costs.

In this paper, we describe and present simulation and experimental performance studies for a form of adaptive self-configuration designed for sensor networks. As we argue in Section 2, these unattended systems will need to self-configure and adapt to a wide variety of environmental dynamics and terrain conditions. These conditions produce regions with nonuniform communication density. We suggest that one of the ways system designers can address such challenging operating conditions is by deploying redundant nodes and designing the system algorithms to make use of that redundancy over time to extend the systems life. In ASCENT, each node assesses its connectivity and adapts its participation in the multihop network topology based on the measured operating region. For instance, a node:

- signals when it detects high packet loss, requesting additional nodes in the region to join the network in order to relay messages,
- reduces its duty cycle if it detects high packet losses due to collisions,
- probes the local communication environment and does not join the multihop routing infrastructure until it is "helpful" to do so.

Why can this adaptive configuration not be done from a central node? In addition to the scaling and robustness limitations of centralized solutions, a single node cannot directly sense the conditions of nodes distributed elsewhere in space. Consequently, other nodes would need to communicate detailed information about the state of their connectivity in order for the central node to determine who should join the multihop network. When energy is a constraint and the environment is dynamic, distributed approaches are attractive and possibly are the only practical approach [22] because they avoid transmitting dynamic state information repeatedly across the network.

Pottie and Kaiser [22] initiated work in the general area of wireless sensor networks by establishing that scalable wireless sensor networks require multihop operation to avoid sending large amounts of data over long distances. They went on to define techniques by which wireless nodes discover their neighbors and acquire synchronism. Given this basic bootstrapping capability, our work addresses the next level of automatic configuration that will be needed to realize envisioned sensor networks, namely, how to form the multihop topology [7]. Given the ability to send and receive packets and the objective of forming an energy-efficient multihop network, we apply

- *The authors are with the Center for Embedded Networked Sensing, Computer Science Department, University of California at Los Angeles, 3531H Boelter Hall, Box 951596, Los Angeles, CA 90095-1596. E-mail: {cerpa, destrin}@cs.ucla.edu.*

Manuscript received 21 Mar. 2004; revised 21 May 2004; accepted 26 May 2004.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMCSI-0111-0304.

well-known techniques from MAC layer protocols to the problem of distributed topology formation.

In the following section, we present a sensor network scenario, stating our assumptions and contributions. Related work is reviewed in Section 3. Section 4 describes ASCENT in more detail. In Section 5, we present some initial analysis, simulation, and experimental results using ASCENT. Finally, in Section 6, we conclude.

2 Distributed Sensor Network Scenario

To motivate our research, consider a habitat monitoring sensor network that is to be deployed in a remote forest. Deployment of this network can be done, for example, by dropping a large number of sensor nodes from a plane or placing them by hand. In this example and in many other anticipated applications of ad hoc wireless sensor networks [5], the deployed systems must be designed to operate under the following conditions and constraints:

- Ad hoc deployment: We cannot expect the sensor field to be deployed in a regular fashion (e.g., a linear array, 2D lattice). More importantly, uniform deployment does not correspond to uniform connectivity owing to unpredictable propagation effects when nodes, and therefore antennae, are close to the ground and other surfaces.
- Energy constraints: The nodes (or at least some significant subset) will be untethered for power as well as communications and therefore the system must be designed to expend as little energy as is possible in order to maximize network lifetime.
- Unattended operation under dynamics: The anticipated number of elements in these systems will preclude manual configuration, and the environmental dynamics will preclude design-time preconfiguration.

In many such contexts, it will be far easier to deploy larger numbers of nodes initially than to deploy additional nodes or additional energy reserves at a later date (similar to the economics of stringing cable for wired networks). In this paper, we present one way in which nodes can exploit the resulting redundancy in order to extend system lifetime.

If we use too few of the deployed nodes, the distance between neighboring nodes will be too great and the packet loss rate will increase or the energy required to transmit the data over the longer distances will be prohibitive. If we use all deployed nodes simultaneously, the system will be expending unnecessary energy at best and, at worst, the nodes may interfere with one another by congesting the channel. In the process of finding an equilibrium, we are not trying to use a distributed localized algorithm to identify a single optimal solution. Rather, this form of adaptive self-configuration using localized algorithms is well suited to problem spaces that have a large number of possible solutions; in this context, a large solution space translates into dense node deployment. Our simulation and experimental results confirm that this is the case for our application.

We enumerate the following assumptions that apply to the remainder of our work:

We assume a Carrier Sense Multiple Access (CSMA) MAC protocol with the capacity to work in promiscuous mode. This clearly introduces the possibilities for resource contention when too many neighboring nodes participate in the multihop network. Our approach should be relevant to TDMA MACs as well because distributed slot allocation schemes will also have degraded performance with increased load.

Our algorithm reacts when links experience high packet loss. The ASCENT mechanism does not detect or repair network partitions of the underlying raw topology. Partitions are more prevalent when node density is low, and our approach is not applicable because, in general, all nodes will be needed to form an effective network. Of course, network partitions can occur even in dense arrays when a swath of nodes are destroyed or obstructed. When such network partitions do occur, complementary system mechanisms will be needed; for example, detecting partitions in the multihop sensor network by exploiting information from long range radios deployed on a subset of nodes and used sparingly because of the power required. We leave such complementary techniques for network partition detection and repair to future work.

The two primary contributions of our design are:

- The use of adaptive techniques that permit applications to ***configure*** the underlying topology based on their needs while trying to save energy to extend network lifetime. Our work does not presume a particular model of fairness, degree of connectivity, or capacity required.
- The use of self-configuring techniques that react to operating conditions ***measured locally***. Our work is not restricted to the radio propagation model, the geographical distribution of nodes, or the routing mechanisms used.

3 Related Work

Our work has been informed and influenced by a variety of other research efforts. There has been a great deal of work in the area of topology control, mostly using theoretical analysis or simulation and involving MAC and power control mechanisms.

There have been several important theoretical evaluations of topology control. Most of this work focuses on the analysis of algorithms for distributed construction of a connected dominating set (CDS) of the corresponding unit-disk graph and the routing strategies using the CDS backbone [11], [29], [1], [12]. Gao et al. [12] present a randomized algorithm for maintaining a CDS with low overhead. Gao et al.'s algorithm assumes the partion of the space in a grid and selects a small number of cluster heads. The total number selected has an approximation factor of $O(\sqrt{n})$ of the minimum theoretically possible. In later work, Gao et al. present a distributed algorithm to construct a restricted Delaunay graph (RDG), where only Delaunay edges with a limited fix transmission radius are included [11]. The work shows that the number of edges in the restricted Delaunay graph is linear in the number of nodes, although the maximum degree of a node may be $\Omega(n)$ in the

worst case. Alzoubi et al. [1] describe a distributed algorithm for constructing a minimum connected dominating set (MCDS) for the unit-disk-graph with a constant approximation ratio of the minimum possible and linear time and message complexity. Wang and Li propose an algorithm to build a geometric spanner that can be implemented in a distributed manner [29]. The node degree is bounded by a positive constant, and the resulting backbone is a spanner for both hops and length.

The above algorithms provide the theoretical limits and bounds of what is achievable with topology control. Our work with ASCENT complements theirs by getting results from experiments using *real* radios, rather than using only simulation and analysis. Recent work [10], [4], [33], [30] evaluating radio connectivity using low-power radios suggests that these radio channels present *asymmetrical links*, *nonisotropic* connectivity, and *nonmonotonic* distance decay of power with distance. It is important to understand the effects that these conditions impose on these topology control algorithms since most of the real conditions observed using real radios violate the assumptions in the previous theoretical studies and may affect correctness. There is poor correlation between the spatial distance and reception rate, so assumptions based on geographic proximity between nodes do not necessarily hold in practice. Furthermore, the radio propagation is *not* circular, presenting nonisotropic properties. Finally, our previous work with SCALE [4] has shown the presence of asymmetric links for 5-30 percent of all pairwise communication, causing serious problems with algorithms that assume bidirectional connectivity.

The main approach followed by *MAC level protocols* to save energy has been to turn off the radios that do not have any scheduled transmission or reception of packets in a particular (usually small) timeframe. These protocols usually trade network delay for energy conservation because of the startup cost associated with turning the radios back on. Sohrabi and Pottie [27] present a self-configuration and synchronization TDMA scheme at the single cluster. This work is more focused on the low-level synchronization necessary for network self-assembly, while we concentrate on efficient multihop topology formation. Sparse Topology and Energy Management (STEM) [26] accepts delays in path-setup time in exchange for energy savings. It uses a second radio (operating at a lower duty cycle) as a paging channel. Sensor-MAC (S-MAC) [32] treats both per-node fairness and latency as secondary to energy conservation. It periodically turns off the radios of idle nodes and uses in-channel signaling to turn off radios that are not taking part in the current communication. More recent work [34] continues to explore MAC-level wake-up schemes. Most of the MAC schemes mentioned above are complementary to our work. ASCENT could establish a particular active topology and then use any of the above mechanisms to gain even further energy savings on the newly created active topology.

Another approach in reducing energy consumption has been to adaptively control the transmit power of the radio. The *lazy* scheduling proposed in Prabhakar et al. [23] transmits packets with the lowest possible transmit power for the longest possible time such that delay constraints are still met. Ramanathan and Rosales-Hain [24] proposed some distributed heuristics to adaptively adjust node transmit powers in response to topological changes caused by mobile nodes. This work assumes that a routing protocol is running at all times and provides basic neighbor information that is used to dynamically adjust transmit power. While power control can be very useful, particularly in asymmetric networks such as cellular telephony, their advantages are less pronounced in sensor networks [4]. Furthermore, the power consumed by these low-power radios in idle state is of the same order of magnitude than the Tx or Rx state, so optimizations on transmit power are less important. Under these conditions, turning the radio off and putting the transceiver in sleep state is essential to extend network lifetime.

In Xu et al. [31], GAF nodes use geographic location information to divide the network into fixed square grids. Nodes in each grid alternate between sleeping and listening, and there is always one node active to route packets per grid. ASCENT does not need any location aids since it is based on connectivity. In addition, geographic proximity may not always lead to radio connectivity; this is why ASCENT uses local connectivity measurements. Chen et al. [6] proposed SPAN, an energy efficient algorithm for topology maintenance, where nodes decide whether to sleep or join the backbone based on connectivity information supplied by a routing protocol. ASCENT does not depend on routing information nor need to modify the routing state; it decides whether to join the network or sleep based on measured local connectivity. In addition, our work does not presume a particular model of fairness or network capacity that the application requires.

Mobile ad hoc networks [16], [20], [21] and directed diffusion [14] adaptively configure the routing or data dissemination paths, but they do not adapt the basic topology. Li and Rus [17] presented a scheme where mobile nodes modify their trajectory to transmit messages in the context of disconnected ad hoc networks. This work may complement ours in case of mobile nodes deployment and in the presence of network partitions.

The adaptive techniques we use were studied extensively to make the MAC layer self-configuring and adaptive more than 20 years ago during the refinement of contention protocols [15], [18]. More recently, SRM [9] and RTCP [25] borrowed these techniques to adaptively adjust parameters such as session message frequency and randomization intervals. In this work, we use those techniques to adapt the topology of a multihop wireless network.

The following section describes the ASCENT protocol in some detail.

4 ASCENT Design

ASCENT adaptively elects "active" nodes from all nodes in the network. Active nodes stay awake all the time and perform multihop packet routing, while the rest of the nodes remain "passive" and periodically check if they should become active.

Consider a simple sensor network for data gathering similar to the network described in Section 2. Fig. 1 shows a simplified schematic for ASCENT during initialization in a

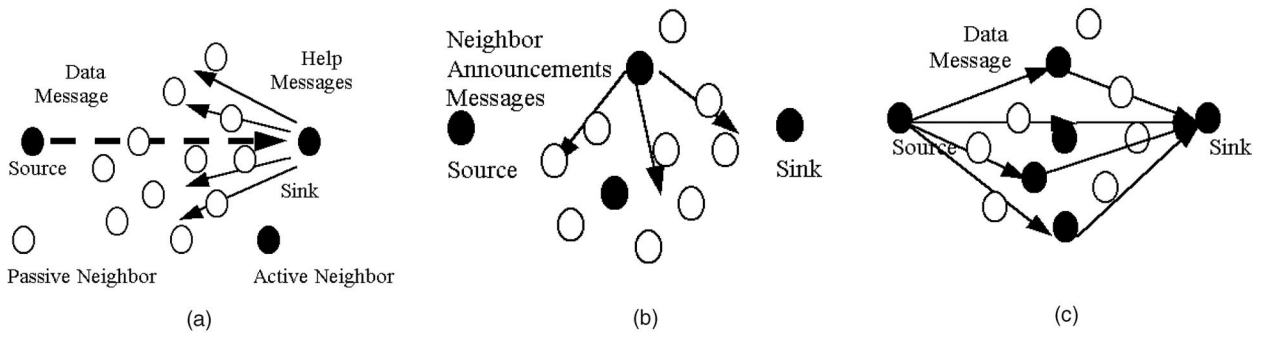


Fig. 1. Network self-configuration example. (a) Communication hole. (b) Transition. (c) Final state.

high-density region. For the sake of clarity, we show only the formation of a two-hop network. This analysis may be extended to networks of larger sizes.

Initially, only some nodes are active. The other nodes remain passively listening to packets but not transmitting. This situation is depicted in Fig. 1a. The source starts transmitting data packets toward the sink. Because the sink is at the limit of radio range, it gets very high packet loss from the source. We call this situation a *communication hole*. The sink then starts sending help messages to signal neighbors that are in listen-only mode—also called *passive neighbors*—to join the network.

When a neighbor receives a *help message*, it may decide to join the network. This situation is illustrated in Fig. 1b. When a node joins the network, it starts transmitting and receiving packets, i.e., it becomes an *active neighbor*. As soon as a node decides to join the network, it signals the existence of a new active neighbor to other passive neighbors by sending an *neighbor announcement message*. This situation continues until the number of active nodes stabilizes on a certain value and the cycle stops (see Fig. 1c). When the process completes, the group of newly active neighbors that have joined the network make the delivery of data from source to sink more reliable. The process will restart when some future network event (e.g., node failure) or environmental effect (e.g., new obstacle) causes packet loss again.

In this section, we describe the ASCENT algorithm and their components. We elaborate on several design choices while we describe the scheme. Our initial analysis, simulations, and experiments in Section 5 focus only on a subset of these design choices.

4.1 ASCENT State Transitions

In ASCENT, nodes are in one of four states: *sleep, passive, test, and active*. Fig. 2 shows a state transition diagram.

Initially, a random timer turns on the nodes to avoid synchronization. When a node starts, it initializes in the *test state*. Nodes in the *test state* exchange data and routing control messages. In addition, when a node enters the *test state*, it sets up a timer T_t and sends *neighbor announcement messages*. When T_t expires, the node enters the *active state*. If, before T_t expires, the number of active neighbors is above the *neighbor threshold* (NT) or if the average *data loss rate* (DL) is higher than the average loss before entering in the *test state*, then the node moves into the *passive state*. If multiple nodes make a transition to the *test state*, then we use the node ID in the announcement message as a tie breaking mechanism (higher IDs win). The intuition behind the *test state* is to probe the network to see if the addition of a new node may actually improve connectivity.

When a node enters the *passive state*, it sets up a timer T_p and sends *new passive node announcement messages*. This information is used by active nodes to make an estimate of the total density of nodes in the neighborhood. Active nodes transmit this density estimate to any new passive node in the neighborhood. When T_p expires, the node enters the sleep state. If, before T_p expires, the number of neighbors is below NT and either the DL is higher than the *loss threshold* (LT) or DL is below the *loss threshold* but the node received a *help message* from an active neighbor, it makes a transition to the *test state*. While in *passive state*, nodes have their radio on and are able to overhear *all* packets transmitted by their active neighbors. No routing or data packets are forwarded in this state since this is a listen-only state. The intuition behind the *passive state* is to gather information regarding the state of the network without causing interference with the other nodes. Nodes in the *passive* and *test states* continuously update the number of active neighbors and data loss rate values. Energy is still consumed in the *passive state* since the radio is still on when not receiving packets. A node that enters the *sleep state* turns the radio off, sets a timer T_s, and goes to sleep. When T_s expires, the node moves into *passive state*. Finally, a node in *active state* continues forwarding data and routing packets until it runs out of energy. If the *data loss rate* is greater than LT, the active node sends *help messages*.

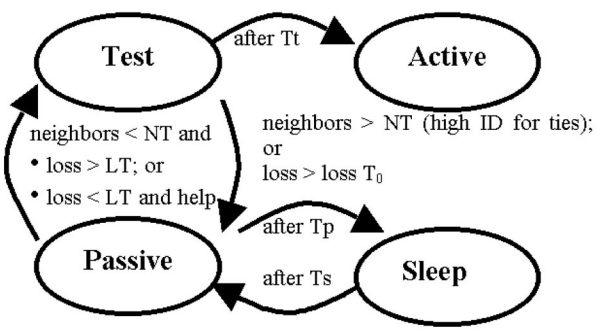


Fig. 2. ASCENT state transitions.

4.2 ASCENT Parameters Tuning

ASCENT has some parameters that can affect its final behavior. In this section, we explain the choices made in the current ASCENT algorithm. A particular application may select different values for some parameters, for instance, trading energy savings for greater reaction time in case of dynamics. ASCENT also provides adaptive mechanisms to dynamically determine the optimal parameter values.

The *neighbor threshold* (NT) value determines the average degree of connectivity of the network. An application could adjust this value dynamically depending on the events occurring in a certain area of the network, for example, to increase network capacity. In this study, we set this value to 4.

The *loss threshold* (LT) determines the maximum amount of data loss an application can tolerate before it requests help to improve network connectivity. This value is very application dependent. For example, average temperature measurements from a sector of a forest will not tend to vary drastically, and the application may tolerate high packet loss. In contrast, tracking of a moving target by the sensor network may be more sensitive to packet losses. In our implementation, this value was set to 20 percent.

The test timer T_t and the passive timer T_p determine the maximum time a node remains in the test and passive states, respectively. They face a similar trade off of power consumption versus decision quality. The larger the timers, the more robust the decision in the presence of transient packet losses (that also affects the neighbor determination), but the greater the power consumed with the radio on, and vice versa. Our work with SCALE [4] has shown that the final determination of these timer values should be dependent on the quality of the reception rate for each link. On the one hand, links that present very high (> 80 percent) or very low (< 20 percent) reception rates show less variability over time and consequently require less time to make an accurate determination of the link quality. On the other hand, links with intermediate reception rates show great variability over time and require more time to make better estimations. We note that it should be possible to design a mechanism that automatically determines the minimum amount of time we should measure the channel to provide some statistical bounds on the accuracy of the link quality estimation, but we have left this as future work. In our implementation, the T_p timer was set to 2 minutes and T_t to 4 minutes.

Similarly, the sleep timer T_s represents the amount of time the node sleeps to preserve energy. The larger the T_s timer, the larger the energy savings, but also the larger the probability of no node in passive state ready to react to dynamics. ASCENT uses an adaptive probabilistic mechanism in order to determine the optimal relationship between the T_p and T_s timers. This mechanism is solely dependent on the average density neighborhood estimate and the probability threshold P_t that a certain k number of nodes in the neighborhood are in the passive state at any given point in time. The details of this mechanism are explained in Section 5.2. In our implementation, the value of k was set to 2 and P_t was set to 95 percent.

4.3 Neighbor and Data Loss Determination

The number of active neighbors and the average data loss rate are values measured *locally* by each node while in passive and test state.

We have chosen to define a neighbor as a node from which we receive a certain percentage of packets over time. This implies having a history window function (CW) that keeps track of the packets received from each individual node over a certain period (time and/or number of messages) and a fixed or dynamic *neighbor loss threshold* (NLS).

In ASCENT, each node adds a unitary monotonically increasing sequence number to each packet transmitted (including data and control packets). This permits neighbor link loss detection when a sequence number is skipped. In addition, we assume application data packets also have some mechanism to detect losses (data payload sequence numbers in our implementation). Additionally, the final packet loss (or its reciprocal reception rate) estimate from each neighbor node is calculated by using an exponentially weighted moving average (EWMA) of the form:

$$EWMA_{current} = \rho \cdot CW + (1 - \rho) EWMA_{previous}.$$

The value of the filter constant ρ was set to $0.\widehat{3}$. This local estimate is periodically exchanged between active and test nodes (not passive nodes) by piggybacking this information in data packets or by sending hello packets in the absence of data traffic.

The number of active neighbors N is defined as the number of neighbors with link packet loss smaller than the neighbor loss threshold (NLS) and with symmetrical links. In our study, we consider a link symmetrical if it has a difference in reception rate of less than 40 percent between the incoming and outgoing reception rate. We have chosen the following formula NLS:

$$NLS = 1 - \frac{1}{N},$$

with N being the number of neighbors calculated in the previous cycle.

When a node gets a neighbor's packet loss estimate larger than the NLS, it no longer considers that node as a neighbor and deletes it from its neighbor list. The intuition behind this formula is the following: As we increase the number of neighbors in the region, the likelihood of any pair of them not listening to each other (or having high losses) increases. Therefore, as we increase the number of neighbors, we should correspondingly increase the neighbor's loss threshold. Not doing so may result in getting a lower neighbor count even though nodes in the region may still interfere with each other. Correspondingly, as we decrease the number of neighbors, we should decrease the neighbor's loss threshold accordingly. (We experimented with some other functions, like an inversely decaying function of 1/N and an exponentially decaying function of 1/N, but the simple formula above worked best).

The average *data loss rate* (DL) is calculated based on the application data packets. Data losses are detected using data sequence numbers. Depending on the routing strategy, a node may receive multiple copies of the same application data packet. We only consider a data loss if the message was

not received from any neighbor during a certain configurable period of time (this allows out of order delivery based on the application needs). Control messages (help, neighbor announcements, and routing) are not considered in this calculation.

4.4 ASCENT Interactions with Routing

ASCENT runs above the link and MAC layer and below the routing layer. All ASCENT control messages are broadcast locally to the neighbors and they do not require any multihop forwarding scheme. ASCENT is not a routing or data dissemination protocol. ASCENT simply decides which nodes should join the routing infrastructure. Ad hoc routing [15], [18], [20], Directed Diffusion [13], or some other data dissemination mechanism then runs over this multihop topology.

ASCENT nodes become active or passive independent of the routing protocol running on the node. In addition, ASCENT does not use state gathered by the routing protocol since this state may vary greatly for different protocols (e.g., ad hoc routing tables and directed diffusion gradients) or require changing the routing state in any way. Currently, if a node is testing the network and it is actively routing packets when it becomes passive, ASCENT depends on the routing protocol to quickly reroute traffic. This may cause some packet loss and, therefore, an improvement that has not been implemented is to inform the routing protocol of ASCENT's state changes so traffic could be rerouted in advance.

We emphasize that, even though we have discussed the ASCENT algorithm in some detail, much experimentation and evaluation of the various mechanisms and design choices is necessary before we fully understand the robustness, scale, and performance of self-configuration. The following section presents our initial findings based on simple analysis, simulation, and an experimental implementation.

5 Performance Evaluation

In this section, we report results from a preliminary performance evaluation of ASCENT. We use simple mathematical models to determine an idealized expected performance of delivery rate, latency, and energy savings as we increase node density. Since our analysis cannot capture the complexity of a full ASCENT scenario, we use simulations and real experiments to further validate the performance evaluation.

5.1 Goals and Metrics

Our goals in evaluating ASCENT were three-fold: first, in order to validate some of the assumptions made during design of the algorithm; perform analysis, simulations, and real experiments; and conduct comparative performance evaluation of the system with and without ASCENT; second, to understand the energy savings and delivery rate improvements that can be obtained by using ASCENT; finally, to study the sensitivity of ASCENT performance to the choice of parameters.

We choose four metrics to analyze the performance of ASCENT: *One-Hop Delivery Rate* measures the percentage of packets received by any node in the network. When all the nodes are turned on—we call this the *Active case*—the packet reception includes all nodes. In the ASCENT case, it includes all nodes but the ones in the sleep state. This metric indicates the effective one-hop bandwidth available to the nodes in the sensor network. *End-to-End Delivery Rate* is the ratio of the number of distinct packets received by the destination to the number originally sent by the source. It provides an idea of the quality of the paths in the network, and the effective multihop bandwidth. A similar metric has been used in ad hoc routing [3]. *Energy Savings* is the ratio of the energy consumed by the Active case to the energy consumed by the ASCENT case. This metric defines the amount of energy savings and network lifetime we gain by using the ASCENT algorithm. Finally, *Average Per-Hop Latency* measures the average delay in packet forwarding in a multihop network. It provides an estimate of the end-to-end delay in packet forwarding.

5.2 Analytic Performance Analysis

To understand the relationship between expected packet delivery and density of nodes, we first use a simple mathematical analysis.

Assume that nodes are randomly distributed in an area A and have an average degree of connectivity of n. Further, assume packets are propagated using flooding with a random back-off upon packet reception. This random component is chosen from a discrete pool of S slots with a uniform probability distribution. Thus, the probability of successfully transmitting a packet with no collisions when there are T potential forwarding nodes in the vicinity is given by:

$$P(success) = \left(\frac{S-1}{S}\right)^{T}. \quad (1)$$

From this formula, we see that, as we increase the density of transmitting nodes T, the probability of successfully delivering packets without collisions decreases proportionally. When all the nodes in the network are able to transmit and receive packets, we find that $T = n$ since every node in the vicinity can transmit packets (assuming a lossless channel, all nodes received the original packet). Increasing the density of nodes increases the probability of collisions in the area. ASCENT fixes the number of transmitters in the area to the *neighbor threshold* (NT) value, resulting in $T = NT$, independent of the total number of nodes, n, deployed. Fig. 3a shows the analytical relation between expected one hop delivery rate versus density of nodes for different S values.

The relation between the hop-by-hop latency introduced by the randomization and the density of nodes can be analyzed similarly. The average latency experienced per hop is related to the number of random slots S and the total number of active nodes T. After reception of a message to be forwarded toward the destination, each of the T active nodes picks a random slot, say $S_1, S_2, \ldots, S_T$. The mean number of all the random slots chosen will tend to be $S/2$

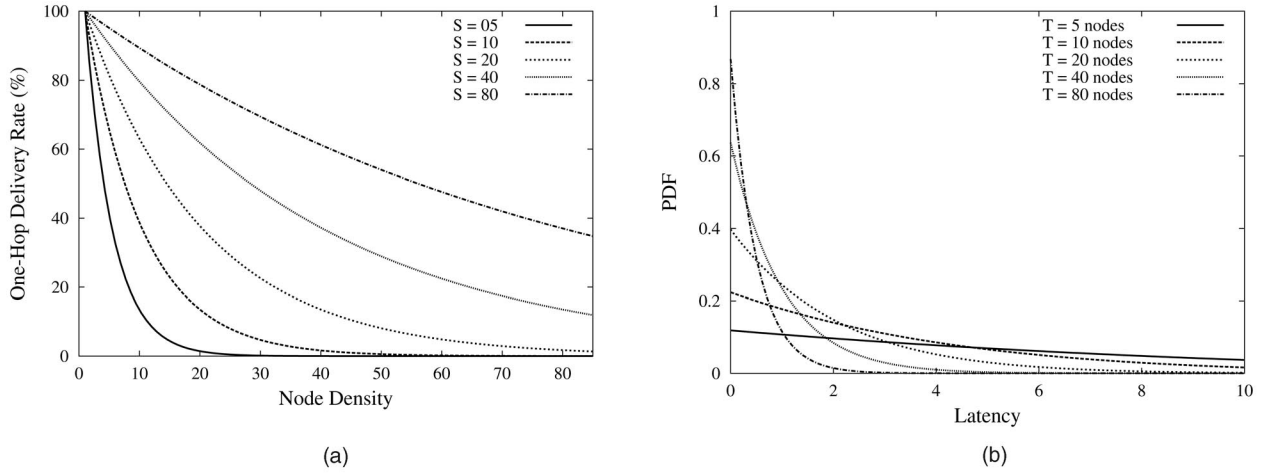


Fig. 3. (a) The expected one hop delivery rate as a function of density. The larger the randomization period, the better the one hop delivery rate for any given density. (b) The probability distribution of the one hop latency. The larger the density, the smaller the probability of a large latency.

since it is a uniform probability distribution. Assuming no collision losses, i.e., $\forall i \neq j \in 1, 2, \ldots, T \Rightarrow S_i \neq S_j$, the hop-by-hop latency is determined by the first message to be forwarded. The delay δ is then:

$$\delta = min(S_1, S_2, \ldots, S_T).$$

We want to find P(δ), the probability distribution of the smaller random time slot picked by T nodes. We define:

$$Q(y) = Prob\lfloor min(S_1, S_2, \ldots, S_T) > y \rfloor min(S_1, S_2, \ldots, S_T) > y \Leftrightarrow \text{ each of } (S_1, S_2, \ldots, S_T) > y.$$

This happens with probability:

$$\left(\frac{S-y}{S}\right)^T therefore\ Q(y) = \left(1 - \frac{y}{S}\right)^T.$$

P(δ) as defined above is:

$$P(\delta) = Q(\delta) - Q(\delta + 1) = \left(1 - \frac{\delta}{S}\right)^T - \left(1 - \frac{(\delta+1)}{S}\right)^T. \quad (2)$$

Fig. 3b shows the P(δ) distribution for different values of T and $S = 20$. When all the nodes in the network are able to transmit and receive packets, we find that $T = n$. As n increases, the mean value of P(δ) decreases. This result corresponds to the intuition that, as we increase the total number of transmitting nodes, the likelihood of any of them picking a smaller random value increases. In the ASCENT case, $T = NT$ independently of the density n, and the mean value of P(δ) remains constant.

Finally, we would like to understand the energy savings that could be obtained by using ASCENT. When the system is not running ASCENT, all the nodes have their radios on, consuming *Idle* power.[1] When the system is running ASCENT, NT nodes have their radios on, while the rest alternate between sleeping and listening. The energy savings (ES) are:

$$ES = \frac{n \cdot Idle}{NT \cdot Idle + (n - NT) \cdot Idle \cdot \frac{T_p}{T_p + T_s} + (n - NT) \cdot Sleep \cdot \frac{T_s}{T_p + T_s}}. \quad (3)$$

The numerator represents the power consumed by all the nodes when not running ASCENT. The denominator represents the power consumed by all nodes running ASCENT. The first term in the denominator indicates the power consumed by the NT nodes selected by ASCENT to have their radios on. The second term in the denominator indicates the energy of nonactive nodes when in passive state, and the third term indicates the energy consumed while in sleep state. We define α to be the ratio of the passive timer T_p to the sleep timer T_s. We also define β to be the ratio of the radio's sleep mode to the idle mode power consumption. By replacing these new definitions in (3), we get:

$$ES = \frac{n}{NT + (n - NT) \cdot \frac{\alpha + \beta}{\alpha + 1}}. \quad (4)$$

Equation (5) shows the upper bound of the energy savings as we increase density.

$$\lim_{n \to \infty} ES = \frac{\alpha + 1}{\alpha + \beta}. \quad (5)$$

Fig. 4 shows the energy savings as we increase the density of nodes for a fixed value of β. For a fixed NT value and a small value of β, as we increase density the power consumption is dominated by the passive nodes in the passive-sleep cycle. The intuition is that the smaller the α, the larger T_s in relation to T_p and, consequently, the larger the energy savings the system can achieve. Note that these savings come at a cost; the larger the T_s, the larger the reaction time of the system in case of dynamics. There is a trade off between the number of nodes we would like in passive state ready to react to dynamics and the energy savings we can achieve by having a more aggressive sleeping schedule. Even if we set up the network with an "optimal" value of α at initialization, the density of the

1. The difference in power consumption between the Tx, Rx, and Idle radio state is not significant. See Section 5.3.

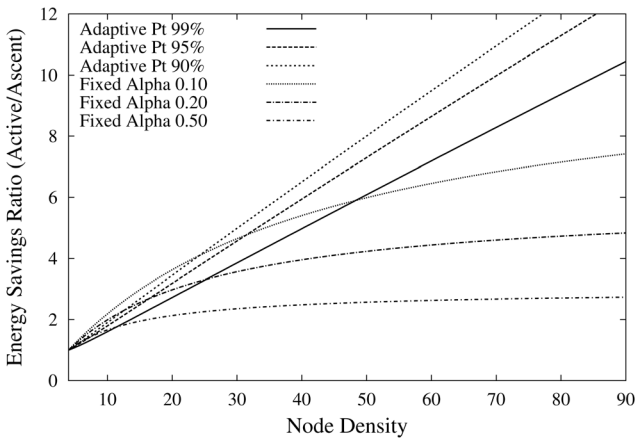


Fig. 4. Energy savings as a function of density for ASCENT fixed and adaptive state timers. Fixed state timers converge asymptotically to a particular maximum value when increasing density. On the other hand, adaptive state timers do not present this asymptotic limit and the energy savings increase linearly as a function of density.

network will not be homogeneous and, even if it is homogeneous initially, it will probably change as nodes drain their batteries and die out. To cope with this trade off, we propose an adaptive probabilistic mechanism where the value of α depends on the minimum probability of k nodes being in passive state at any given moment in time and the density of nodes in the neighboring region.

If we assume that nodes alternate between passive and sleep state and that their schedules are independent, the probability of any node being in passive or sleep state is given by:

$$P(passive) = \frac{\alpha}{\alpha+1} \qquad P(sleep) = \frac{1}{\alpha+1}. \tag{6}$$

We want to find the minimum value of α such that the probability of at least k nodes being passive at any given moment in time is larger than a minimum probability threshold P_t. We call this value α_{P_tk}. Any α value smaller than α_{P_tk} will not comply with the minimum probability requirement, and any α value larger than α_{kP_t} will comply with the minimum probability requirement but will expend unnecessary energy.

For the given state probabilities given in (6), the probability of at least k nodes in passive state at the same time is given by:

$$P(k) = 1 - \left(\frac{1}{\alpha+1}\right)^n \cdot \frac{\alpha^k - 1}{\alpha - 1} \tag{7}$$

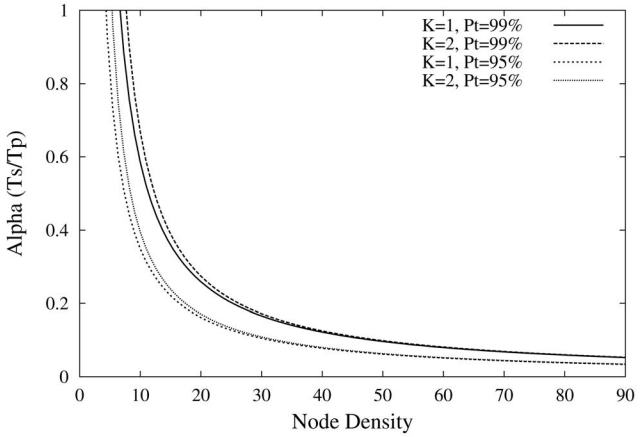
The Appendix shows the proof of (7). We want to find the values of α for different values of k, $P(k)$ being equal to the minimum probability threshold P_t. For $k = 1$ (at least one passive node at any given time) and $k = 2$ (at least two passive nodes at any given time), there are closed-form solutions to the value of α (α_{P_t1} and α_{P_t2}):

$$\alpha_{P_t1} = 10^{-\frac{1}{n}\cdot\log(1-P_t)} - 1, \tag{8}$$

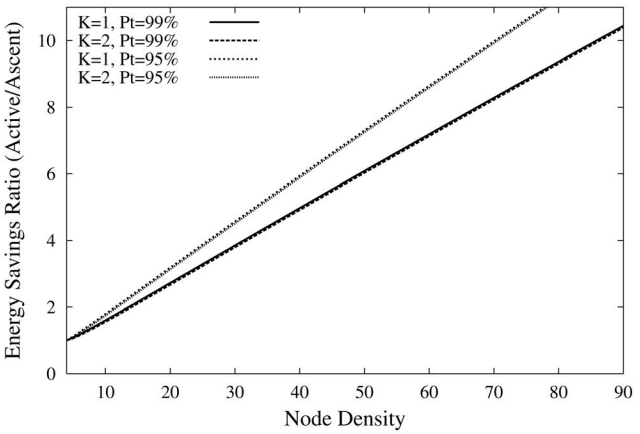
$$\alpha_{P_t2} = 10^{\frac{1}{1-n}\cdot\log(1-P_t)} - 1. \tag{9}$$

For other values of k, the optimal value α_{P_tk} for a specific P_t and density n can be found by using iterative numerical techniques for the solution of nonlinear equations (e.g., Newton's method). The initial searching value x_0 could be set to α_{P_t2}. Fig. 5a shows the optimal values of α for $k = 1$ and $k = 2$ for different probability thresholds.

Fig. 4 shows the energy savings of ASCENT with adaptive state timers. In this case, we no longer have an asymptotic behavior as density increases like in the previous fixed timers case. The energy savings increase linearly with density, and the slope of the line is primarily determined by the probability threshold P_t (see below). Fig. 5b shows that the impact of additional redundancy by incrementing the value of k (concurrent passive nodes) has less of an impact and only reduces the energy savings by a minimal factor.



(a)



(b)

Fig. 5. (a) The optimal α values for $k = 1$ and $k = 2$ for different probability thresholds. (b) The energy savings ratio as a function of density for different values of k and P_t. It is clear from the graphs that the probability threshold P_t has the most noticeable effect in the determination of the value of α and the energy savings ratio.

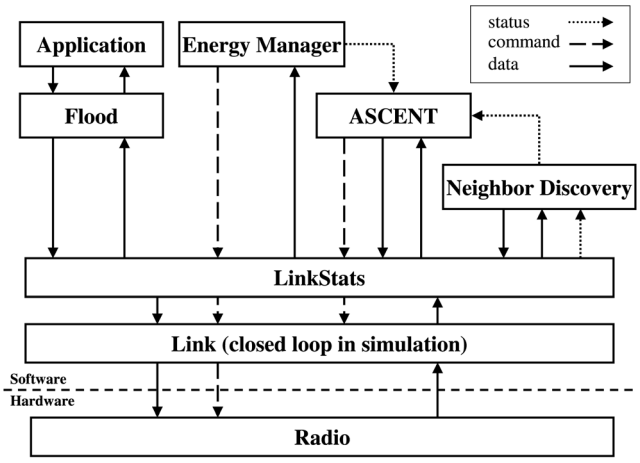


Fig. 6. ASCENT code structure. ASCENT was developed in a modular way so other developers could reuse as much functionality as possible, even when not running the ASCENT topology control algorithm.

5.3 Simulation and Experimental Methodology

5.3.1 Implementation

ASCENT implementation was developed using the EmStar programming environment [13]. We implemented ASCENT using a number of fine-grained modules. Fig. 6 shows the diagram of the code structure. The first is the *LinkStats* module, which adds a monotonically increasing sequence number to each packet sent by any module on the node. It monitors packets arriving from other nodes and maintains detailed packet statistics without increasing channel use (but slightly reducing the maximum data payload). This module also implements the exponentially weighted moving average (EWMA) filter for the reception rate of each neighbor. The second module is *Neighbor Discovery*, which sends and receives heartbeat messages and maintains a list of active neighbors. Third, to evaluate energy usage, we created the *Energy Manager* module that acts as a simulated battery for each node. It counts packets sent and received, idle time, and radios powering on and off; energy is deducted from an initial supply accordingly. Finally, we created the *ASCENT* protocol implementation itself, which uses the information provided by the other modules.

5.3.2 Simulator

ASCENT was simulated using the built in simulator (*emsim*) provided by EmStar [13]. The simulator essentially runs exactly the *same* code base as the implementation, with no modifications. As in reality, the nodes must interact using their radios and are not allowed to share state directly. Instead of using real radios and sensors, *emsim* provides a channel simulator that models the behavior of the environment. The channel model used in our simulations is a statistical model based on extensive radio connectivity traces gathered when developing earlier versions of ASCENT and SCALE [4]. The simulator is also able to provide CSMA style of collisions.

5.3.3 Experimental Testbed

Fig. 7 shows pictures of the hardware components of our testbed. The ceiling array [13] used in the experiments is composed of various serial port multiplexors attached to a testbed PC. Fig. 7b shows an image of the ceiling array deployed in our lab. We use UTP Cat 5 cables of different lengths (up to 30 meters) and attach one end of the cable to the multiplexor and the other end to a node. A total of 55 nodes are used in the testbed. The nodes are wall powered.

Fig. 7a shows a picture of the Mica 1, the node used in the experimental testbed. Table 1 shows the main features of the hardware platform used.

5.3.4 Scenarios and Environment

In order to study the performance of ASCENT's algorithms as a function of density, we run experiments with different densities ranging from 5 to 40 nodes. In this study, density is defined topologically, i.e., the density of nodes is defined by the average degree of connectivity of all the nodes in the experiment and *not* by their physical location (geographical density). Since we could not easily change the location of nodes in the ceiling array and, since the physical size of our lab is limited, we achieved different levels of density by adjusting the transmit power of the RF transceiver. The

(a) (b)

Fig. 7. Experimental Testbed. The ceiling array is composed of a PC attached to various serial multiplexors. Several UTP cables run from each multiplexor to the deployment locations in the ceiling where a mote is attached at the end. (a) Mica 1 mote. (b) Indoor office, UCLA CENS lab ceiling array.

TABLE 1
Node Characteristics

	Mica 1
CPU Processor	Amtel 128
Prog. Memory (KB)	128
Data Memory (KB)	4
Serial RS232	needs adapter
Clock Speed (MHZ)	4
RF Manufacturer	RFM [19]
RF Transceiver	TR1000
Radio frequency (MHz)	916
Modulation	ASK
Throughput (kbps)	13.3
TX power [0dBm] (mW)	< 1
Antenna	Omni whip

average number of hops in the topologies obtained by this method was three. All the experiments were done in an indoor environment, with obstacles such as furniture, walls, cubicles, doors, etc. The simulations replicate the same scenarios tried in the experiments. For each simulation, we vary the density of nodes from 5 to 80 nodes. In addition, for larger multihop simulations, we incremented the number of sources and destinations from 1 to 5. The average number of hops in the simulations was six. In all the experiments and simulations, the source(s) and the destination(s) were placed at the edge of the network to maximize the number of hops and usage of transit nodes (nodes transmitting traffic from/to the source/destination). Each experimental point in the graphs presented in the following sections is the average of three experimental trials, and each simulation point in the graphs represents the average of five simulated trials. All the results include confidence intervals with a degree of confidence of 95 percent.

5.3.5 Traffic

In each experiment, one source sends approximately 200 messages with temperature and light sensor readings (the readings were stored values). The data rate was set to three sensor reading messages per minute. In each simulation, one or more sources send approximately 400 messages each. The data rate was the same as the experiments. In all our experiments and simulations, we operate the sensor network far from overload. Hence, our sensor nodes do not experience congestion. In spite of experimenting with uncongested networks, our nodes can incur packet losses due to dynamics and interference.

5.3.6 Routing

We use flooding as our routing protocol. In order to reduce contention when multiple nodes try to reforward packets received at the same time, the flooding module has a programmable randomization interval. Upon receiving a packet, the flood module will wait for a random time between zero and the maximum randomization interval. In our experiments, the randomization interval was set to 5 seconds (unless otherwise noticed). The choice of flooding routing for our experiments was due to several reasons. First, we lacked other routing implementations (e.g., AODV) for our experimental platform, so, instead, we chose flooding because its simplicity avoids complications due to the specifics of the routing protocol. Second, several routing algorithms still use some form of flooding as part of their routing strategy (e.g., dissemination of interests in directed diffusion or routing state with link state protocols). Finally, several applications in sensor network require propagating information to all nodes (e.g., trigger alert notifications, user query dissemination), and flooding is the simplest mechanism to achieve it.

5.3.7 Energy Model

To model the energy consumption, we looked at the manual specifications of the RFM Tx-1000 [19]. We found the values for Tx:Rx:Idle:Sleep in mW 36:9:9:0.015 for the RFM. Several studies [6], [28] have reported differences on the order of 10:1 between Tx/Rx/Idle and Sleeping power consumption for 802.11 wireless LAN cards. For the low power radios we study, this difference is on the order of 100:1. This relation is important since it is the β factor defined in the previous section. The small differences in energy consumption between the Tx, Rx, and Idle states coupled with the big difference with Sleep state implies that differences in radio traffic (caused for example by different routing strategies) are not the dominant factor in terms of energy savings. The time the nodes keep their radios on, switching periodically to off, is more important since there are more than two orders of magnitude difference between Tx/Rx/Idle and Sleep states power consumption. In our model, we did not consider the energy consumed by the CPU.

The remainder of this section presents our simulation and experimental results.

5.4 Network Capacity

Our first simulations and experiments compare the one-hop delivery rate and the end-to-end delivery rate of the system with and without ASCENT (with adaptive timers enabled).

Fig. 8a shows the one-hop delivery rate as a function of the density in a multihop network. The "No-collisions" curve shows the average one-hop delivery rate in the network for the different densities tested. It shows the average losses due to environmental effects in the absence of simultaneous transmissions. The values were obtained by running SCALE [4] on the ceiling array testbed with different transmission power values. The results are encouraging. To a first degree, there are no important differences between the expected analytical and simulated performance and the performance using real radios up to densities of 40 nodes. In the Active case (no self-configuration, all nodes are turned on), all the nodes join the network and forward packets. This case has low delivery rate because, as we increase the density of nodes, the probability of collisions increases accordingly when using flooding as a routing strategy. It rapidly reaches around 40 percent with densities of 20 nodes and enters into a saturation region after that. ASCENT limits the number of active nodes to the

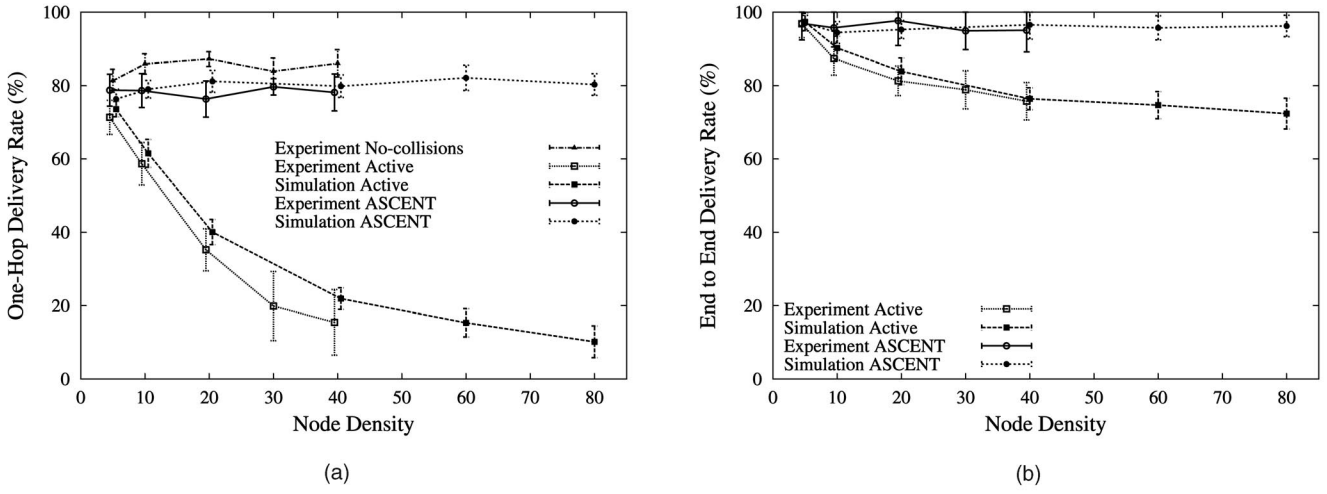


Fig. 8. (a) The one-hop delivery rate as a function of density. ASCENT limits the number of active nodes forwarding traffic to NT and reduces contention for the channel. (b) The end-to-end delivery rate as a function of density. ASCENT end-to-end delivery rate is stable for the range of densities tested.

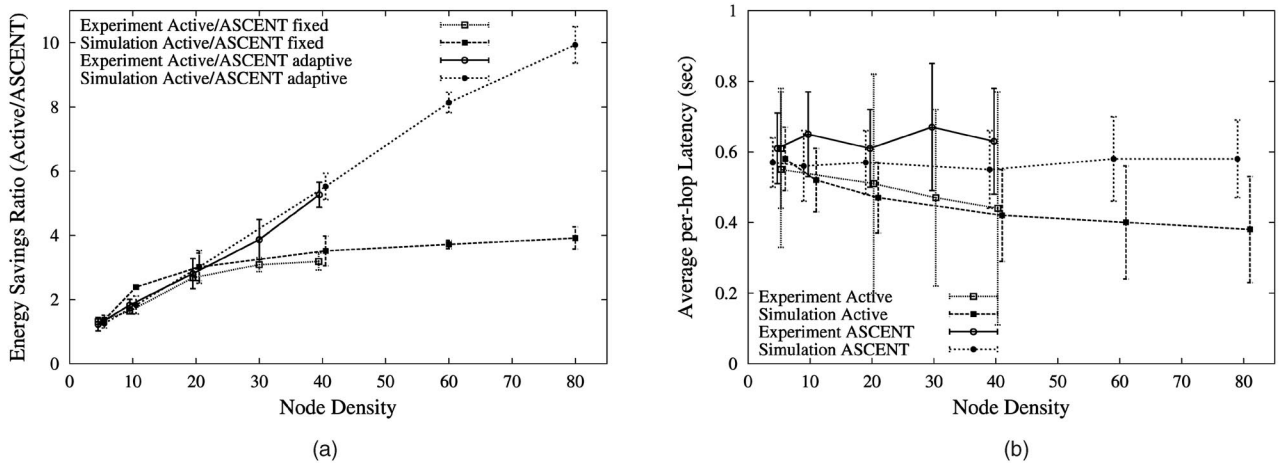


Fig. 9. (a) The energy savings ratio as a function of density. ASCENT provides a significant amount of energy savings over the Active case, up to a factor of 4 with fix timers and 10 with adaptive timers for high density scenarios. (b) The average per-hop latency as a function of density. ASCENT slightly increases the average hop by hop latency.

NT value and, therefore, does not increase channel contention with larger densities.

Fig. 8b shows the end-to-end delivery rate, i.e., the percentage of packets transmitted by the source that reached the destination. In the experiments, each packet traverses an average of three hops. The simulations were done on a larger network, with packets traversing, on average, six hops from source to destination. We can see that ASCENT outperforms the Active case. ASCENT's performance remains stable as the density increases, which demonstrates the scalability properties of our algorithms as the density increases. The Active case does not perform as bad as one would expect based on the one-hop delivery rate shown in Fig. 8a. This is because the end-to-end delivery ratio metric only requires that at least *one* copy of the original packet sent by the source reach the destination. Even in a high-density environment with high losses due to contention for the channel, the likelihood of receiving *one* copy of the packet is still high using flooding.

The results in this section clearly favor ASCENT because they compare the worst-case scenario of flooding contention with increased density (in particular for one-hop delivery rate). We expect comparable performance results between ASCENT and the Active cases when using nonflooding based routing strategies. Evaluating the performance of ASCENT with different routing strategies is an area of future work.

5.5 Energy Savings

This section evaluates ASCENT's ability to save energy and increase network lifetime.

In these experiments and simulations, we did not consider the energy spent by the source(s) or the destination(s). For the real experiments, the values are not direct measurements of energy consumption but indirect measurements using the time the nodes spent in the different ASCENT's states.

Fig. 9a shows the average energy consumption ratio between the active and ASCENT cases as a function of density. We present results using two versions of the

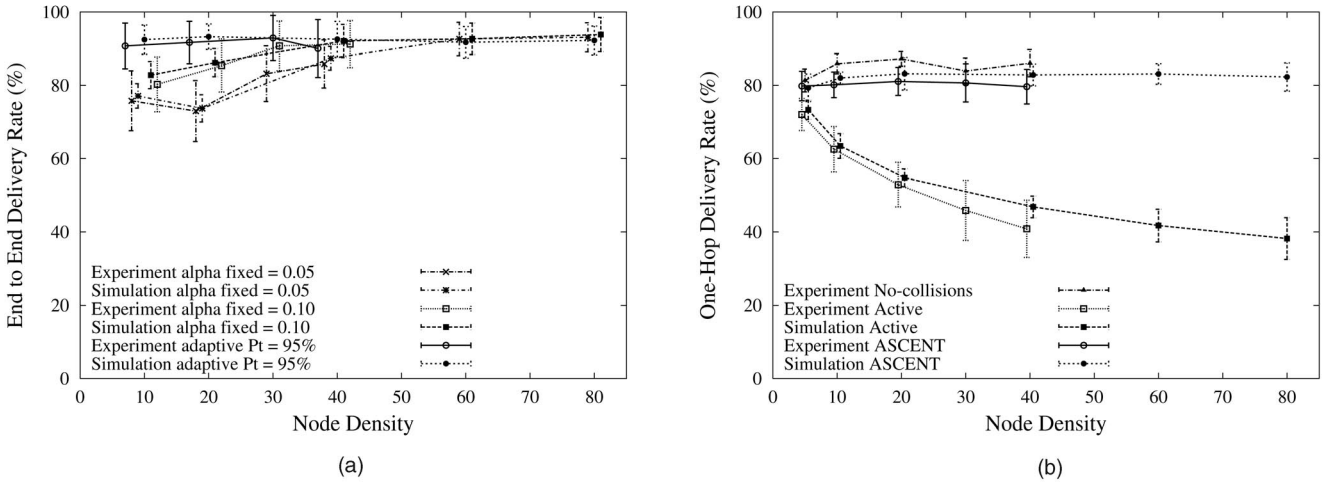


Fig. 10. (a) The end-to-end delivery rate as a function of density for fixed and adaptive state timers. By using fixed state timers, ASCENT reaction to dynamics performance at lower densities may decrease. When using adaptive state timers, ASCENT performance is stable for the range of densities tested. (b) The one-hop delivery rate as a function of density for a larger flooding randomization interval (10 seconds). ASCENT provides better delivery rates independently of the randomization interval.

ASCENT algorithm, one with fixed and the other with adaptive state timers. From these results, we find that ASCENT provides a significant amount of energy savings over the Active case.

When using ASCENT with fixed state timers, we find that, as density increases, energy savings do not increase proportionally. This result may seem counterintuitive because, in ASCENT, the number of active nodes remains constant as density increases, and one would expect to save more energy as the fraction of active nodes decreases. From the analysis shown in Section 5.2, we see that the energy consumption, as we increase density, is dominated by the passive-sleep cycle of the passive nodes and not by the energy consumed by the fraction of active nodes. ASCENT provides a factor of 4 in energy savings in this case.

When using ASCENT with adaptive state timers, we find that, as density increases, energy savings do increase proportionally. In this case, nodes can be more aggressive in their sleeping cycle when they detect a high density region. Also, note that the level of aggressiveness can be tuned based on the probabilistic guarantees offered to react to dynamics.

In both cases (fixed and adaptive), the performance in simulation and real experiments is qualitatively similar, but below the expected performance based on the analytical results. The main reason for this is that the analysis done in Section 5.2 does not consider losses from the environment which induce ASCENT to increase the number of nodes with the radio on to maintain a usable topology and, consequently, reduce the energy savings in practice.

5.6 Latency

In this section, we study ASCENT's impact on packet delivery latency.

Fig. 9b shows the average per-hop latency as a function of density. Note that we consider only packets that successfully reach the destination in the results (successful end-to-end delivery). We use the average per-hop density to compensate between the different number of hops between the experiments and simulations.

We can see from the graph that ASCENT increases the average per-hop latency when compared to the active case. When using flooding as the routing strategy, the end-to-end delay is affected by the amount of randomization introduced at each hop and the number of nodes forwarding the packets. When density increases, the active case reduces the average per-hop latency because there is a larger probability of a node picking a smaller random interval to forward the packet when there are more forwarding nodes, as was shown in Section 5.2. ASCENT fixes the number of nodes able to forward packets independently of density and, consequently the average per-hop latency tends to remain stable for the same randomization interval.

The reduction in latency for the active case is not as big as predicted in the analytical model. The reason for this is simple: In practice, when considering losses due to the environment and contention for the channel, a packet forwarded fast may not always reach destination, and the average delay per hop can increase from the ideal.

5.7 Reaction to Dynamics

In this section, we evaluate how ASCENT reacts to dynamics introduced by node failures in the active topology.

For these experiments, we let the system run until a stable topology is in place. We then manually kill a set of active nodes such that there is a network partition between the source(s) and destination(s) in the *active* topology.

Fig. 10a shows the end-to-end delivery rate for ASCENT with both fixed and adaptive state timers. The conditions of the experiment are identical to the experiments performed in Section 5.4. The values have been slightly moved on each density point to improve readability. We can see that, for the fixed values of α we tested, the end to end delivery rate does not decreases much at high densities. This is because there is high probability that a passive node in the neighborhood exists to fix the *communication hole*. As density decreases, the performance of ASCENT with fixed state timers also decreases. This is because, for certain fixed values of α, it is possible that we are being overaggressive in

saving energy and all nodes in the neighborhood might be sleeping at the time of the active topology failure. ASCENT with adaptive state timers is more stable for the range of densities we tested.

5.8 Sensitivity to Parameters

This section evaluates the sensitivity of the ASCENT algorithm to the choice of randomization values used in the flooding routing.

Fig. 10b shows the one-hop delivery rate as a function of density for a larger randomization interval used in flooding. For this experiment, we picked a randomization interval of 10 seconds. Fig. 8a shows a similar graph for a randomization interval of 5 seconds. When comparing the two graphs, we can clearly see that, for larger randomization intervals, we get an increase in the average one-hop delivery rate for different densities. However, there is a trade off since larger randomization intervals increase the end-to-end latency. For the different levels of randomization we tried, the ASCENT case always outperforms the Active case, even when the former has a smaller randomization interval than the latter.

The increase is important for the Active case, but it is only marginal for ASCENT. This is because ASCENT operates on a reduced topology independently of the actual density of nodes and increasing the randomization interval does not help much. This also shows that the expected performance of ASCENT is more stable independently of the choice of the randomization interval.

6 Conclusion and Future Work

In this paper, we described the design, implementation, analysis, simulation, and experimental evaluation of ASCENT, an adaptive self-configuration topology mechanism for distributed wireless sensor networks.

There are many lessons we can draw from our preliminary experimentation. First, ASCENT has the potential for significant reduction of packet loss and increase in energy efficiency. Second, ASCENT mechanisms were responsive and stable under systematically varied conditions.

Furthermore, our paper reports on results from experiments using *real* radios, demonstrating the importance of self-configuring techniques that react to the operating conditions *measured locally*.

In the near future, we will evaluate the interactions of ASCENT with new MAC mechanisms and the use of robust statistical techniques to improve online link quality estimation. We will also investigate the use of load balancing techniques to distribute the energy load and explore the use of wider area links to detect network partitions. More generally, we are interested in understanding the relationships between topology control mechanisms, like ASCENT, and different routing strategies.

This work is an initial foray into the design of self-configuring mechanisms for wireless sensor networks. Our distributed sensing network simulations and experiments represent a nontrivial exploration of the problem space. Such techniques will find increasing importance as the community seeks ways to exploit the redundancy offered by cheap, widely available microsensors, as a way of addressing new dimensions of network performance such as network-lifetime.

Appendix

Proof of the Probability of k Passive Nodes Equation

Given a set of n nodes that alternate between passive and sleep states with probabilities given by (6), we would like to find the probability $P(k)$ of at least k passive nodes at any given moment in time:

$$\begin{aligned} &P(\text{at least } k \text{ passive nodes}) \\ &= 1 - P(\text{at most } k-1 \text{passive nodes}), \end{aligned}$$

for k = 1:

$$\begin{aligned} P(1) &= 1 - P(0 \text{ nodes passive}) \\ &= 1 - \left(\frac{1}{\alpha+1}\right)^n, \end{aligned}$$

for k = 2:

$$\begin{aligned} P(2) &= 1 - (P(0 \text{ nodes passive}) + P(1 \text{ node passive})) \\ &= 1 - \left[\left(\frac{1}{\alpha+1}\right)^n + \left(\frac{1}{\alpha+1}\right)^{n-1} \cdot \left(\frac{\alpha}{\alpha+1}\right)\right], \end{aligned}$$

generalizing for any k:

$$\begin{aligned} P(k) &= 1 - (P(0 \text{ nodes passive}) + P(1 \text{ nodes passive}) + \\ &\quad \cdots + P(k-1 \text{nodes passive})) \\ &= 1 - \left[\left(\frac{1}{\alpha+1}\right)^n + \left(\frac{1}{\alpha+1}\right)^{n-1} \cdot \left(\frac{\alpha}{\alpha+1}\right) + \right. \\ &\quad \left. \cdots + \left(\frac{1}{\alpha+1}\right)^{n-k+1} \cdot \left(\frac{\alpha}{\alpha+1}\right)^{k-1}\right] \\ &= 1 - \left[\left(\frac{1}{\alpha+1}\right)^n \cdot \alpha^0 + \left(\frac{1}{\alpha+1}\right)^n \cdot \alpha^1 + \right. \\ &\quad \left. \cdots + \left(\frac{1}{\alpha+1}\right)^n \cdot \alpha^{k-1}\right] \\ &= 1 - \left(\frac{1}{\alpha+1}\right)^n \cdot (\alpha^0 + \alpha^1 + \cdots + \alpha^{k-1}) \\ &= 1 - \left(\frac{1}{\alpha+1}\right)^n \cdot \frac{\alpha^k - 1}{\alpha - 1}. \end{aligned}$$

Acknowledgments

This work was made possible with support from The Center for Embedded Networked Sensing (CENS) under US National Science Foundation Cooperative Agreement CCR-0120778 and the SCOWR program funded by the US National Science Foundation under grant ANI-9979457.

References

[1] K.M. Alzoubi, P.-J. Wan, and O. Frieder, "Message-Optimal Connected-Dominating-Set Construction for Routing in Mobile Ad Hoc Networks," *Proc. Third ACM Int'l Symp. Mobile Ad Hoc Networking and Computing (MobiHoc)*, June 2002.

[2] B. Badrinath, J. Scholtz, M. Srivastava, K. Mills, and V. Stanford, *IEEE Personal Comm.,* special issue on smart spaces and environments, K. Soolins, ed., Oct. 2000.
[3] J. Broch, D.A. Maltz, D.B. Johnson, Y. Hu, and J. Jetcheva, "A Performance Comparison of Multihop Wireless Ad Hoc Network Routing Protocols," *Proc. Fourth Ann. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MobiCom '98),* pp. 85-97, Oct. 1998.
[4] A. Cerpa, N. Busek, and D. Estrin, "SCALE: A Tool for Simple Connectivity Assessment in Lossy Environments," Technical Report CENS Technical Report 0021, Center for Embedded Networked Sensing, Univ. of California, Los Angeles, Sept. 2003.
[5] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, "Habitat Monitoring: Application Driver for Wireless Communications Technology," *Proc. SIGCOMM Workshop Comm. in Latin America and the Carribean,* Apr. 2001.
[6] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *Proc. Seventh Ann. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MobiCom),* pp. 85-96, July 2001.
[7] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks," *Proc. Fifth Ann. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MobiCom),* pp. 263-270, Aug. 1999.
[8] *Comm. ACM,* special issue on embedding the internet, D. Estrin, R. Govindan, and J. Heidemann, eds., vol. 43, no. 5, May 2000.
[9] S. Floyd, V. Jacobson, C.-G. Liu, S. McCanne, and L. Zhang, "A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing," *Proc. ACM Special Interest Group on Data Comm. (SIGCOMM),* pp. 342-356, Aug. 1995.
[10] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker, "Complex Behavior at Scale: An Experimental Study of Low-Power Wireless Sensor Networks," Technical Report UCLA CSD-TR 02-0013, Center for Embedded Networked Sensing, Univ. of California, Los Angeles, and Intel Research Lab, Univ. of California, Berkeley, Feb. 2002.
[11] J. Gao, L.J. Guibas, J. Hershburger, L. Zhang, and A. Zhu, "Geometric Spanner for Routing in Mobile Networks," *Proc. Second ACM Symp. Mobile Ad Hoc Networking and Computing (MobiHoc 01),* pp. 45-55, Oct. 2001.
[12] J. Gao, L.J. Guibas, J. Hershburger, L. Zhang, and A. Zhu, "Discrete and Computational Geometry," *Proc. Second ACM Symp. Mobile Ad Hoc Networking and Computing (MobiHoc 01),* vol. 30, no. 1, pp. 45-65, 2003.
[13] L. Girod, J. Elson, A. Cerpa, N. Ramanathan, T. Stathopoulos, and D. Estrin, "Emstar: A Software Environment for Developing and Deploying Wireless Sensor Networks," *Proc. 2004 USENIX Technical Conf.,* June-July 2004.
[14] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *Proc. Sixth Ann. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MobiCom),* Aug. 2000.
[15] V. Jacobson, "Congestion Avoidance and Control," *Proc. ACM Special Interest Group on Data Comm. (SIGCOMM),* pp. 314-329, Aug. 1988.
[16] D.B. Johnson and D.A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," *Mobile Computing,* T. Imielinski and H. Korth, eds., pp. 153-181, 1996.
[17] Q. Li and D. Rus, "Sending Messages to Disconnected Users in Disconnected Ad Hoc Mobile Networks," *Proc. Sixth Ann. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MobiCom),* pp. 44-55, Aug. 2000.
[18] R. Metcalfe and D. Boggs, "Ethernet: Distributed Packet Switching for Local Computer Networks," *Comm. ACM,* pp. 395-404, July 1976.
[19] RFM Monolithics, "Tr1000 Low Power Radio System," http://www.rfm.com/products/data/tr1000.pdf, 2003.
[20] V. Park and S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," *Proc. 18th Ann. Joint Conf. IEEE Computer and Comm. Soc. (INFOCOM),* pp. 1405-1414, Apr. 1997.
[21] C. Perkins and E. Royer, "Ad Hoc on Demand Distance Vector (AODV) Routing," *Proc. Second IEEE Workshop Mobile Computing Systems and Applications,* pp. 90-100, Feb. 1999.
[22] G. Pottie and W. Kaiser, "Embedding the Internet: Wireless Integrated Network Sensors," *Comm. ACM,* vol. 43, no. 5, pp. 51-58, May 2000.
[23] B. Prabhakar, E. Uysal-Biyikoglu, and A.E. Gamal, "Energy-Efficient Transmission over a Wireless Link Via Lazy Packet Scheduling," *Proc. 20th Ann. Joint Conf. IEEE Computer and Comm. Soc. (INFOCOM),* pp. 386-394, Apr. 2001.
[24] R. Ramanathan and R. Rosales-Hain, "Topology Control of Multihop Wireless Networks Using Transmit Power Adjustment," *Proc. 19th Ann. Joint Conf. IEEE Computer and Comm. Soc. (INFOCOM),* pp. 404-413, Mar. 2000.
[25] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real Time Applications," Request for Comments (RFC 1889), Jan. 1996.
[26] C. Schurgers, V. Tsiatsis, and M. Srivastava, "Stem: Topology Management for Energy Efficient Sensor Networks," *Proc. IEEE Aerospace Conf.,* pp. 78-89, Mar. 2002.
[27] K. Sohrabi and G. Pottie, "Performance of a Novel Self-Organization Protocol for Wireless Ad Hoc Sensor Networks," *Proc. IEEE Vehicular Technology Conf.,* Sept. 2000.
[28] M. Stemm and R. Katz, "Measuring and Reducing Energy Consumption of Network Interfaces in Hand-Held Devices," *IEICE Trans. Comm.,* vol. 80, no. 8, pp. 1125-1131, Aug. 1997.
[29] Y. Wang and X.-Y. Li, "Geometric Spanners for Wireless Ad Hoc Networks," *Proc. 22nd Int'l Conf. Distributed Computing Systems (ICDCS 2002),* July 2002.
[30] A. Woo, T. Tong, and D. Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks," *Proc. First ACM Sensys Conf.,* pp. 14-27, Nov. 2003.
[31] Y. Xu, J. Heidemann, and D. Estrin, "Geography-Informed Energy Conservation for Ad Hoc Routing," *Proc. Seventh Ann. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MobiCom),* pp. 70-84, July 2001.
[32] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. 21st Ann. Joint Conf. IEEE Computer and Comm. Soc. (INFOCOM),* pp. 1567-1576, June 2002.
[33] Y.J. Zhao and R. Govindan, "Understanding Packet Delivery Performance in Dense Wireless Sensor Networks," *Proc. First ACM Sensys Conf.,* Nov. 2003.
[34] R. Zheng, J.C. Hou, and L. Sha, "Asynchronous Wakeup for Ad Hoc Networks," *ACM Int'l Symp. Mobile Ad Hoc Networking and Computing,* June 2003.

Alberto Cerpa received the MSc degree in computer science from the University of Southern California (USC) in 2000, the MSc degree in electrical engineering from USC in 1998, and the engineer degree in electrical engineering from the Buenos Aires Institute of Technology, Argentina in 1995. He is a PhD candidate in computer science at the University of California at Los Angeles, working under the supervision of Professor Deborah Estrin in the Center for Embedded Networked Sensing (CENS). His interests lie broadly in the computer networking and distributed systems areas. His recent focus has been on systems research in wireless sensor networks, with emphasis on network self-configuration, radio channel measurement and characterization, programming models, and development of wireless testbeds. He is a student member of the IEEE.

Deborah Estrin received the PhD degree in computer science from the Massachusetts Institute of Technology (1985), and the BS degree in electrical engineering and computer engineering from the University of California at Berkeley in 1980. She is a professor of computer science at the University of California at Los Angeles and the director of the Center for Embedded Networked Sensing (CENS), a US National Science Foundation Science and Technology Center awarded in 2002. She was on the faculty of Computer Science at the University of Southern California from 1986 through mid-2000. She has been instrumental in defining the national research agenda for wireless sensor networks, first chairing a 1998 DARPA ISAT study and then a 2001 NRC study. Her research group develops algorithms and systems to support robust networks of physically embedded devices with a particular focus on environmental monitoring. She is a fellow of the ACM, IEEE, and AAAS.