

LODE: A Low Delay Sensor Network System

CS261 Final Report

Dario Vlah

January 14, 2002

1 Introduction

Ad hoc wireless networks are envisioned to provide connectivity in situations where no infrastructure is available. These networks consist of untethered nodes which communicate over wireless radio links, and use batteries as a source of power. Ad hoc networking research has been focused on two domains: dealing with mobility, and improving energy efficiency.

Mobile ad hoc networks are intended for situations where node movement is frequent. For example, soldiers in a field, rescue teams at sites of accident, or colleagues at a meeting without an access point may use mobile ad hoc routing to communicate while moving around. The routing protocols proposed in the literature typically use frequent beacon messages to detect mobility, and use specialized techniques to deal with broken routes, such as local route repair.

In contrast, nodes in ad hoc sensor networks are stationary; the emphasis lies on maximizing the lifetime of the network. In an example scenario where many cheap sensors are scattered randomly across an area of interest, the sensors remain quiet until required to deliver measurements (some low-rate communication is needed, such as for discovering adjacent nodes). The quiet period may last very long compared to the time of activity; for example, aircraft may work on scattering the sensors over the course of several days, only to have the network answer requests of duration on the order of seconds. It is important that the sensors use their energy reserves sparingly during the quiet period, or otherwise the network may not be functional long enough to be useful. This paper is motivated primarily by sensor networks; thus, we will focus our efforts on energy efficiency, and forego issues that arise from mobility.

Duty cycling is an effective technique for saving energy in radio communication. It consists of putting wireless interfaces to sleep some large fraction of the time, instead of keeping them turned on and consuming power. It is important to note that wireless interfaces consume power even while listening for incoming transmissions; thus, an inactive interface can neither transmit nor receive. Adjacent sensor nodes must wait for scheduled wakeup periods in order to continue

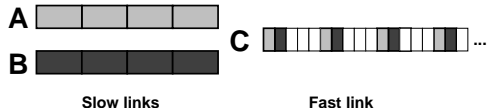


Figure 1: Illustration of Time Division Multiplexing

their conversation.

A consequence of duty cycling are increased communication delays. While this is acceptable for periodic management messages that usually occur in the background, it should not limit data traffic. A sensor network should have the ability to achieve high data rates when needed, while remaining as quiet as possible during the periods of inactivity. We developed *LODE*, a sensor network system which achieves both of these goals—using long duty cycles to control power consumption, and a novel scheduling mechanism to achieve low delay for data traffic.

The paper is organized as follows. We discuss some related work in Section 2. In Section 3 we describe the *LODE* system and its scheduling mechanism. We evaluate the system’s performance and compare it to two other systems in Section 4, and present our conclusions in Section 5.

2 Related Work

Much work has been done on mechanisms that schedule communication at discrete time slots. Most generally, such mechanisms serve two purposes: multiplexing and medium access.

The first category is usually termed *Time Division Multiplexing (TDM)*, and allows digital switches to combine data from several slow links and transmit it over a single fast link by interleaving it in time. For example, in Figure 1, two slow links $A - C$ and $B - C$ merge at the switch C , and insert data into the fast link at preassigned time slots.

The second category is known as *Time Division Multiple Access (TDMA)*, and permits several network nodes to gain efficient access to a shared medium such as a wire or the air. In TDMA, each node is assigned a different time slot during which it has sole control of the medium. The key advantage of TDMA is freedom from collisions; however, this comes at the expense of overhead due to managing time slot reservations and synchronizing the clocks at the participating nodes. For example, usually there is a special node, such as a base station, which maintains time slot reservations. Such a node in Bluetooth networks [3] is called a *master*, and can arbitrate communication with up to 7 other nodes, called *slaves*.

Several recent papers suggested topology control as an effective energy sav-

ing mechanism in *dense* networks, by activating only nodes which are essential to maintain a fully connected network. Cerpa and Estrin [1] proposed and implemented an adaptive protocol which maintains a constant density of active nodes in the network. In their system, nodes switch between active and inactive state depending on how successful the network is in delivering messages. The event of more than a certain threshold fraction of messages getting lost is interpreted as a signal that active nodes are spaced too far apart; consequently, inactive nodes join the topology in order to fill in the gaps.

In sensor networks, two new research directions in energy efficiency arose recently: *in-network processing*, and *nested control*. The first case relies on computations which turn raw sensor data into smaller amounts of resulting data. For example, suppose that a border node *A* in Figure 2 requests a total count of the sensors in the network. When individual counts from nodes *B* and *C* meet at *D*, *D* can add them up and send the resulting sum which takes up much less space than the original two reports, achieving significant energy savings. An in-network processing system was simulated by Heinzelman et al. [5], while Heidemann et al. [4] measured the benefits of in-network processing on a small sensor network testbed.

The second direction explores the inefficiencies of nested queries. For example, suppose node *A* in Figure 2 desires to track positions of objects moving through the sensor network. Computing the positions is hard, because it requires triangulation, and thus at least three nodes must sense the object. Node *A* can obtain the position in an inefficient way using two queries: first, ask all sensors to report detecting an object using a cheap, imprecise method like a motion detector. Then, obtain precise readings from several sensors in the vicinity of the report. Using the precise readings *A* can compute the exact position using methods such as blind beamforming proposed by Yao et al. [10]. This sequence of steps is inefficient because the information required to decide to initiate the second query, and the information *asked for* by the second query both originate in one area of the network, while the decision is made in another, possibly distant area. Thus, some of the signaling messages in these two queries make unnecessarily long trips through the network.

In a nested query system, node *A* would request and receive only the final result—the exact position of the observed object; the intermediate query would be executed locally around the object. A straightforward way to design the control would be to get the sensors around the object to form a cluster and

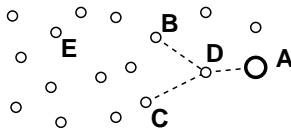


Figure 2: Sensor network example

elect a clusterhead which would perform the computations on raw data, and send a result to node *A*. The paper by Heidemann et al. [4] mentioned earlier evaluated the benefits of nested queries on their sensor testbed.

3 Design Description

The primary means of conserving energy in LODE sensor nodes is the TDMA transmission mechanism.

This section describes the components of the LODE sensors, listed in Figure 3. We start by first introducing the generic ones—shown white—and then devote a separate subsection to the components new to LODE—shown gray in the figure.

The MAC layer arbitrates access to the radio medium by dividing time into a number of discrete time slots, and storing the state of each slot in a table. Most importantly, the radio interface is powered down during unused slots; otherwise, the radio may be set to await incoming transmissions or send queued data. The time slot table is managed by components above the MAC layer.

The neighbor discovery mechanism is similar to a protocol which appeared in [8]. The sensor nodes initially announce themselves, and listen for others' announcements without interruption. The nodes record the announcements by assigning the current time slot to the originating neighbor. Nodes gradually reduce the time spent listening in favor of powering off the radio; we will call this remaining listening time the *control period*. Neighbors maintain their relationship through infrequent beacon messages, which they send during the control period.

The request routing protocol component receives requests and maintains reverse paths to destinations which we will call *base stations*. We envision base stations to be special nodes much more capable than the sensors; such nodes would be responsible for assigning tasks or collecting data from the network. The requests occur during the control period.

The data source module represents the sensing functionality of the node, such as temperature, pressure, magnetic field sensors, infrared motion detectors, etc.

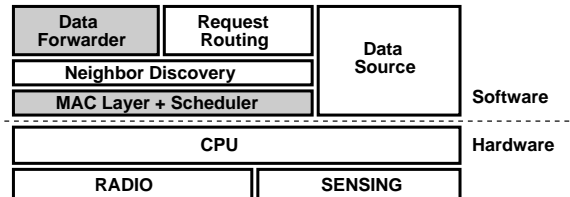


Figure 3: Components of a sensor node

3.1 Data Scheduling and Forwarding

A generic TDMA mechanism introduces delay into data forwarding because messages spend time waiting for designated outgoing time slots; this can be minimized by scheduling the outgoing time slots close to the arrival. Time slot scheduling is the key feature of LODE that distinguishes it from a generic TDMA system.

The scheduling occurs in two steps. First, a request message from a base station propagates through the network, eventually reaching a sensor node that matches the request. Then, this node allocates a time slot outside of the control period, and returns a path setup message indicating the time slot. The upstream node allocates a time slot to receive the originator’s transmissions, as well as an additional time slot following as close as possible for the purpose of relaying the packet. The path setup message continues toward the base station, leaving behind a sequence of minimally separated time slots.

Should the path setup fail, an error message is bounced back to the originator, tearing down the allocated time slots. The resources are eventually timed out, in the event that error messages get lost.

A minor difficulty with allocating transmission time slots is that the corresponding reception time at the upstream node must be free. Thus, all nodes must learn the transmission schedules of their upstream neighbors before making scheduling decisions. An appropriate time to announce these schedules is in the first step, while request messages propagate through the network.

Data forwarding is easy given a properly setup path—the originating node need just emit data packets during appropriate time slots. The packets need not have any header fields but length; addressing is not required because the destination is implied by the time of the transmission.

4 Evaluation

We first describe the experimental testbed, and then present the experimental results.

4.1 Testbed

4.1.1 Description of Sensor Nodes

Our target sensor platform is a microcontroller driven system with a minimal amount of resources, sensing capabilities, and a low-power radio. We choose this platform for two reasons. First, it has barely enough computation power required to perform sensing and wireless communication; thus, it supports the required functionality at nearly minimal energy cost. Second, even though it is likely that Moore’s law will bring about more powerful devices of the same size,

and thus allow more generous operating software, it is also true that the size of the networked sensors with just the basic capabilities will keep decreasing.

We chose the ATMEL AVR 8515 8-bit microcontroller as our target CPU. This processor has modest resources; it runs at several MHz, while providing 4K bytes of ROM, and 512 bytes of RAM. This processor was previously used to power several sensor node prototypes, described in [6], as a target for a security protocol [7], and one of the experimental platforms in [4].

We chose to perform our experiments using simulation. To this end, we developed a simulated wireless device using a simple radio propagation model, and combined it with an AVR CPU simulator. Together, the simulated sensor nodes are similar to those reported in previous papers.

A disadvantage of fine-grained simulations such as this is the performance which is necessarily worse than coarse-grained network simulators like ns [2] or Glomosim [9]. However, there are three important advantages. First, we gain the ability to measure energy consumption of the sensor nodes much more precisely. Second, the programming environment retains the constraints of the sensor platform, allowing us to gain additional insight into the feasibility of various algorithms we consider applying in the sensor network; the security protocol described by Perrig et al. [7] is a good example of how resource constraints can guide system design. Third, a system developed on the simulator can be expected to run on physical sensor nodes with minimal changes. Though we have not been able to confirm this yet, we expect the major differences to arise in dealing with the radio interface.

We track energy expenditure using power consumption values listed in Table 1; these values are given for the processor clock of 4 MHz, and the radio bit rate of 2.4 Kbps. To put these figures into perspective, we note that readily available 802.11 wireless network adapters draw roughly a hundred times more power, and communicate a hundred times faster.

The processor speed limits the maximum bit rate our system can sustain. In particular, our radio processing is driven by a timer which fires every 665 clock cycles, and requires three interrupts per bit. At 4MHz, we achieve a bit rate of 2 Kbps.

Radio Power Consumption (mW)		CPU Power Consumption (mW)	
RX Power	5.4	Active Mode	9.0
TX Power	36.0	Sleep Mode	3.0
Off Power	0.015	Power-down Mode	0.003

Table 1: Power consumption figures for an Atmel AVR 8515 microcontroller, and RF Monolithics TR 1000 radio transceiver, provided by the manufacturers.



Figure 4: Network topology used in the experiments. An edge between two nodes indicates that they are within radio range. The leftmost node generates the data, and the rightmost node receives it.

4.1.2 Description of the Network Topology and Traffic Load

The network topology, shown in Figure 4, consists of 10 nodes arranged in a straight line, and spaced far enough apart that each node is within reach of just a single node on either side.

This topology is loaded with a constant rate stream of messages generated at one end of the chain, and destined to the opposite end. The messages consist of a destination identifier, and a sequence number; together with other protocol headers, they number 8 bytes. One message is generated every 25 seconds.

4.2 Experiments

The experiments evaluate the LODE system in two areas: total expended energy, and average delay experienced by data messages. We compare the results with two other systems: a standard TDMA system, and a generic wireless Ethernet system (denoted *Generic* hereon); alone, the two systems offer lowered energy consumption and low delay, respectively.

Each of the three systems ran for 2000 seconds. At each node, we recorded total expended energy, total energy consumed by the processor, and the energy expended on communication. We measured delay as the time needed for messages to travel from the source node (1) to a destination node.

The LODE system and the standard TDMA system both used 48 time slots, each covering a one second period. Thus, there is an 48 second delay between repeating time slots.

4.2.1 Energy Consumption

Figure 5 shows the energy consumption history for the radio at the leftmost node. We can clearly see that the generic system draws a steady amount of power, while the two energy-efficient systems exhibit a sharp decline around the 250 second mark—at this time nodes stop listening for new neighbors, and start turning off their radios during but a few time slots.

Figure 6 shows the energy consumption history for the CPU at the leftmost node. This time, we see that the three systems spend similar amounts of energy on processing.

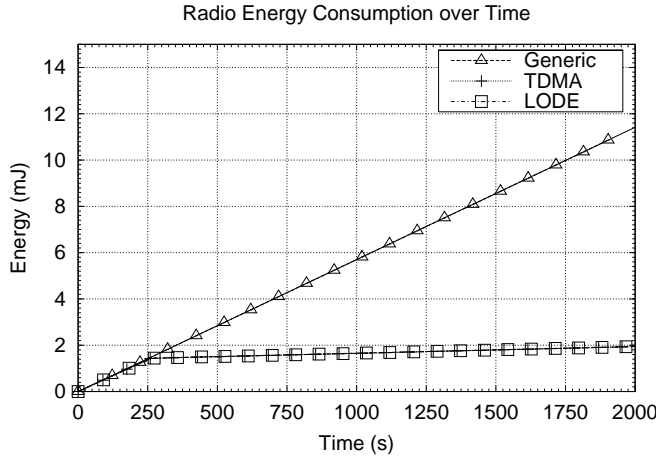


Figure 5: Radio energy consumption over the lifetime of the network

4.2.2 Delay

Consider all messages successfully received at the destination node 10. For each such message, we record the times at which it passed through the intermediate nodes. Figure 7 reports the average of these times for each of the tree systems. Since the number of messages successfully delivered by each system was different, the figure uses error bars to allow a proper comparison.

Starting from the top, we see that the TDMA system imposes the most delay, because each intermediate hop adds an equal amount of delay to related messages. As a result, the total path delay is dependent on the number of intermediate nodes in the path.

The next system is our LODE. We can see that at the start of the path, LODE introduces a delay similar to TDMA, because messages generated at approximately random times can not be guaranteed to always appear exactly during the outgoing time slot. However, once the messages get under way, the delay imposed by the rest of the path is almost non-existent; thanks to time slot scheduling, the intermediate nodes are able to relay the messages as soon as they arrive.

Finally, the generic system exhibits the best delay performance. The reason is clear—the system does not manage time as a resource, but instead instantly relays all received messages.

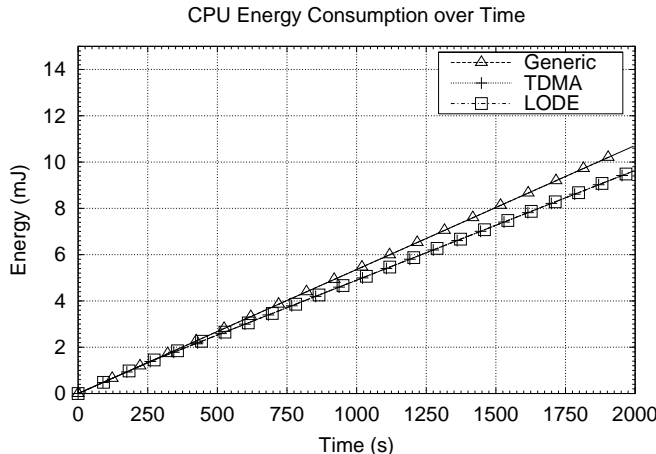


Figure 6: CPU energy consumption over the lifetime of the network

5 Conclusions

The key conclusion we draw is that time-slot scheduling can be an effective technique for improving delays in TDMA systems. As the results show, the LODE sensor network system of this paper achieves lower traffic delay than earlier systems, while retaining a comparable level of energy efficiency.

We note that the results were obtained on a relatively simple network topology. In order to obtain a stronger conclusion which finds time-slot scheduling effective in a larger set of network topologies and traffic patterns, a more extensive evaluation is needed. For now this remains a goal for future work.

The simulation environment of this paper consists of an accurate microcontroller emulator, which allowed us to make precise energy measurements. We believe this environment will prove useful for future work on sensor network systems.

References

- [1] A. Cerpa and D. Estrin. Ascent: Adaptive self-configuring sensor network topologies. Technical Report UCLA/CSD-TR 01-0009, UCLA Computer Science Department, 2001.
- [2] K. Fall and K. Varadhan. *ns notes and documentation*. The VINT project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, Nov. 1997. Available from <http://www-mash.cs.berkeley.edu/ns/>.

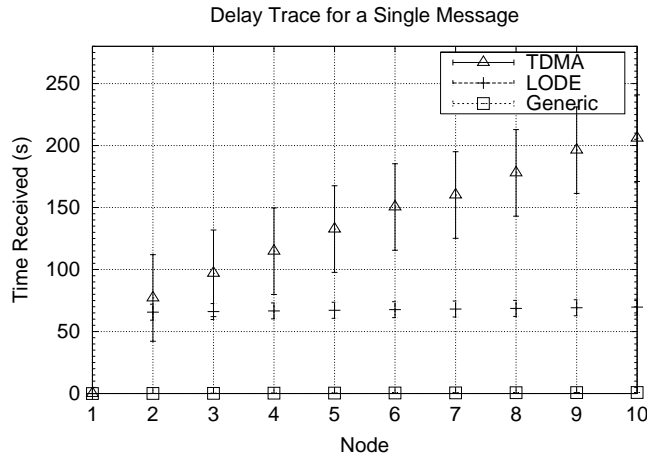


Figure 7: The plot shows average times required for messages to appear at each node in the network. Error bars indicate the uncertainty of the data points

- [3] J. Haartsen, M. Naghshineh, J. Inouye, O. J. Joeressen, and W. Allen. Bluetooth: Vision, goals, and architecture. *Mobile Computing and Communications Review*, 4(2):38–45, Oct. 1998.
- [4] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan. Building efficient wireless sensor networks with low-level naming. In *18th ACM Symposium on Operating Systems Principles (SOSP 2001)*, Banff, Canada, Oct. 2001.
- [5] W. R. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Hawaii International Conference on System Sciences*, Maui, Hawaii, Jan. 2000.
- [6] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX)*, Cambridge, MA, USA, Nov. 2000.
- [7] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. Spins: Security protocols for sensor networks. In *Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (ACM MobiCom)*, pages 189–199, Rome, Italy, July 2001.
- [8] K. Sohrabi, J. Gao, V. Ailawadhi, and G. Pottie. A self organizing wireless sensor network. In *37th Annual Allerton Conference on Communication, Control, and Computing*, Sept. 1999.
- [9] UCLA Parallel Computing Laboratory and Wireless Adaptive Mobility Laboratory. *GloMoSim: A Scalable Simulation Environment for Wireless and Wired Network Systems*. <http://pcl.cs.ucla.edu/projects/domains/glomosim.html>.

- [10] K. Yao, R. E. Hudson, C. W. Reed, D. Chen, and F. Lorenzelli. Blind beamforming on a randomly distributed sensor array system. *IEEE Journal on Selected Areas in Communications*, 16(8):1555–67, Oct. 1998.