# Wave Scheduling and Routing in Sensor Networks

NIKI TRIGONI
Birkbeck College, University of London
and
YONG YAO, ALAN DEMERS, and JOHANNES GEHRKE
Cornell University
and
RAJMOHAN RAJARAMAN
Northeastern University

Sensor networks are being increasingly deployed for diverse monitoring applications. Event data are collected at various sensors and sent to selected storage nodes for further in-network processing. Since sensor nodes have strong constraints on their energy usage, this data transfer needs to be energy-efficient to maximize network lifetime. In this article, we propose a novel methodology for trading energy versus latency in sensor database systems. We propose a new protocol that carefully schedules message transmissions so as to avoid collisions at the MAC layer. Since all nodes adhere to the schedule, their radios can be off most of the time and only wake up during well-defined time intervals. We show how routing protocols can be optimized to interact symbiotically with scheduling decisions, resulting in significant energy savings at the cost of higher latency. We demonstrate the effectiveness of our approach by means of a thorough simulation study, using synthetic data as well as real-world traffic workloads.

This work was supported in part by an Alfred P. Sloan Fellowship; by Grants IIS-00121175, IIS-0133481, and IIS-0330201 from the National Science Foundation of the United States; by Grant AFOSR F49620-02-1-0233 from the Air Force Office of Scientific Research of the United States; the Cornell Information Assurance Institute; and by the British Engineering and Physical Sciences Research Council (EPSRC award EP/C547640/1). Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.
Authors' addresses: N. Trigoni, School of Computer Science and Information Systems, Birkbeck, University of London, Malet Street, London WC1E 7HX, United Kingdom; email: niki@dcs.bbk.ac.uk; Y. Yao, 4156 Upson Hall, Cornell University, Ithaca, NY 14853; email: yao@cs.cornell.edu; A. Demers, 4115A Upson Hall, Cornell University, Ithaca, NY 14853; email: ademers@cs.cornell.edu; J. Gehrke, 4105B Upson Hall, Cornell University, Ithaca, NY 14853; email: johannes@cs.cornell.edu; R. Rajaraman, College of Computer and Information Science, 202 WVH, Northeastern University, Boston, MA 02115; email: rraj@ccs.neu.edu.

Categories and Subject Descriptors: C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*Distributed databases*; C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Routing protocols*

General Terms: Algorithms

Additional Key Words and Phrases: Energy conservation, node scheduling, sensor networks

**ACM Reference Format:**

Trigoni, N., Yao, Y., Demers, A., Gehrke, J., and Rajaraman, R., 2007. Wave Scheduling and Routing in Sensor Networks, ACM Trans. Sens. Netw. 3, 1, Article 2 (March 2007), 35 pages. DOI = 10.1145/1210669.1210671 http://doi.acm.org/10.1145/ 1210669.1210671

## 1. INTRODUCTION

Sensor networks consisting of small nodes with sensing, computation, and communication capabilities are becoming ubiquitous. A powerful paradigm that has emerged recently views a sensor network as a distributed SensorDBMS and allows users to extract information by injecting declarative queries in a variant of SQL. In deploying a SensorDBMS one should consider important limitations of sensor nodes on computation, communication, and energy consumption. Energy is the most valuable resource for unattended battery-powered nodes. Since radio communication consumes most of the available energy, SensorDBMSs need energy-efficient data-dissemination techniques in order to extend their lifetime.

An important communication pattern within sensor networks is the sending of sensor readings from the nodes where they are generated, referred to as *sources*, to designated sensor nodes, referred to as *sinks*. For example, consider a heterogeneous sensor network with two types of sensor nodes: many small-scale *source nodes* with low-power multi-hop communication capabilities, and a few powerful *gateway nodes* connected to the Internet. In this setup, data flows from the sources to the gateway nodes. Another example is motivated by resource savings through *in-network processing*. In-network processing algorithms coordinate data collection and processing in the network at designated nodes, which act as data sinks [Ratnasamy et al. 2002; Ghose et al. 2003]. Since energy is a major resource constraint, we would like this data flow between sources and sinks to be as energy-efficient as possible; in particular, for non-time-critical applications, we would like to trade message latency versus energy usage as events are routed from the nodes where they originated to the sinks.

In order to achieve energy-efficient data flows between sources and sinks, we address several challenges intrinsic to ad hoc network communication: minimizing collisions at the MAC layer, managing radios in a energy-efficient manner, and selecting energy-efficient routes. In this article we consider data dissemination strategies that avoid collisions (and message retransmissions) at the cost of higher message latency. We carefully coordinate transmissions between nodes, allowing them to turn their radios off most of the time. The current generation of radios consume nearly as much power when listening or receiving as when transmitting (typical idle:receive:transmit ratios are 1:1:2.7 [Crossbow Mica2 2005], 1:1.2:1.7 [Chen et al. 2002], 1:2:2.5 [Kasten 2001], and

1:1.05:1.4 [Stemm and Katz 1997]). Thus, the ability to turn them off when not needed yields significant energy savings.

This article is organized as follows: Section 2 introduces the problem and presents complexity results under an idealized model. Section 3 presents a realistic radio model that we adopt for designing our protocols. Section 4 and 5 describe our scheduling and routing algorithms, respectively. An experimental evaluation of the proposed algorithms is presented in Section 6. Section 7 discusses related work and Section 8 concludes the article. This work extends Trigoni et al. [2004] with formal proofs of problem complexity (Section 2), with modifications of the original wave and routing algorithms to cope with radio irregularities and localization errors (Sections 4 and 5), and with experiments conducted using a realistic radio model, real sensor workloads, and a variety of network topologies (Section 6).

## 2. PROBLEM DEFINITION AND COMPLEXITY RESULTS

In this section, we give a formal problem definition, and we show results about the complexity of the problem under ideal, unrealistic radio assumptions to get an understanding of the hardness of the problem.

With coordinated scheduling, a data dissemination protocol has two components: a *scheduling algorithm* that activates network edges so that their transmissions do not interfere with one another, and a *routing algorithm* that selects routes for individual messages. Two important performance metrics are *energy consumption* and *message delay*. In this section, we consider each of these metrics and sketch complexity results for the following optimization problems: (i) finding an optimal pair of routing and scheduling algorithms; (ii) finding an optimal routing algorithm for a given schedule; (iii) finding an optimal schedule for a given collection of routes.

The underlying framework for our optimization problems is as follows. We assume the sensor nodes form a multi-hop wireless network embedded in the plane. For the purpose of complexity analysis, we assume that the node radios have identical ranges of one unit. Thus, the nodes form a *unit disk graph:* two nodes are connected by an edge if and only if the Euclidean distance between them is at most 1. This simplifying assumption about the radio model is used only for complexity analysis purposes. We refer to nodes that locally generate messages as *source nodes* (or *sources*) and to the final destinations of messages as *sink nodes* (or *sinks*). We represent the communication workload by the rate of message generation at each source $i$, given by $r_i$, together with a probability distribution $p_{ij}$, giving the probability that a message generated at source $i$ is destined for sink $j$.

**Energy minimization.** In the *energy minimization problem*, we are given a communication workload among the sources and the sinks, and our goal is to determine a data dissemination scheme that minimizes the energy consumed in delivering all messages within a bounded delay. In our model, we assume that the energy consumed when a network edge is activated is $(\alpha + \beta m)$, where $\alpha$ is a fixed *start-up cost* for turning the radio on, $\beta$ is the per-message transmission and reception cost, and $m$ is the number of messages sent during the activation.

THEOREM 2.1. *For any $\alpha > 0$ and $\beta > 0$, finding an energy-optimal routing-scheduling pair that satisfies a given delay bound is NP-hard, even when there is only one sink.*

PROOF. We first establish the NP-hardness of finding an energy-optimal routing-scheduling pair that satisfies a given delay bound. Our proof is by a reduction from the rectilinear Steiner tree (RST) problem. In the RST problem, we are given a set of points in the plane, and we seek a minimum length rooted rectilinear tree connecting the origin to the points. A tree is rectilinear if every edge of the tree is aligned with the X- or Y-axis. It is known [Garey and Johnson 1977] that RST is NP-complete even when the given points and the Steiner tree points are constrained to lie on a grid.

We reduce RST to this problem by letting the origin be the lone sink, the grid points be all the sensor nodes, and the RST points be the source nodes. We set the traffic pattern as follows. At each time step, a message, destined for the sink, is generated at each source independently with probability $\varepsilon\alpha/(\beta kn)$, where $n$ is the number of nodes, $k$ is the number of sources, and $\varepsilon < 1$. Let $P \geq n^2$ denote a bound on the delay of any message. Consider the schedule in which messages are routed every $P - n$ steps along a minimum Steiner tree $T$ connecting the source nodes to the sink. This schedule satisfies the delay constraint and incurs an average energy cost of at most $(|T| + \varepsilon)P/(P - n)$ every $P$ steps. On the other hand, consider any schedule that guarantees that every message is delivered within $P$ steps. During every interval of $P$ steps, this schedule needs to activate edges in some subgraph $G$ that connects the source nodes to the sink; the energy cost during this interval is at least $|G|$. By setting $\varepsilon$ sufficiently small and $P$ sufficiently large, we can ensure that $(|T| + \varepsilon)P/(P - n) < |G|$, for any $G$ that is larger than the minimum Steiner tree. Thus, an optimal schedule will yield a minimum Steiner tree for RST, completing the desired reduction. □

This theorem leads to two natural questions. First, what is the complexity of computing an energy-optimal activation schedule *given a fixed set of routes*? For general graphs, we can show that the preceding problem is NP-hard, using a reduction from the NP-complete minimum feedback vertex set problem [Garey and Johnson 1979]. Second, we consider the problem of finding a set of energy-optimal routes *given an activation schedule*. In this case, since the start-up cost is determined entirely by the activation schedule, the total energy consumption can be minimized in polynomial time by routing every message along a min-hop path.

**Delay minimization.** In the *delay minimization problem*, we are given a communication workload and seek a data dissemination protocol that minimizes maximum message propagation delay. It is already known that minimizing delay in an *ad hoc* wireless network is NP-hard even for the special case where nodes exchange messages only with their neighbors [Sen and Huson 1996]. This reduction can be extended to unit disk graphs.

THEOREM 2.2. *Finding a routing-scheduling pair that minimizes maximum delay is NP-hard.*

Proof. The proof is by a reduction from the 3-coloring problem in unit-disk graphs, which is known to be NP-hard [Clark et al. 1990]. Given an $n$-node unit-disk graph $G$ of the 3-coloring instance, we introduce a new graph $G'$, which consists of two copies of $G$, one copy translated by a distance of $\varepsilon < \Delta$ from the other. The communication workload involves every node of $G$ sending a message to its copy, at the rate of 1/3. The delay bound for every message is 3.

We now argue that all the messages can be delivered within a delay of 3 units if and only if $G$ admits a 3-coloring. Clearly, if $G$ does have a 3-coloring, then we can activate the nodes according to the colors and obtain a schedule of period 3 that delivers every message within 3 units. In the other direction, any periodic activation schedule that delivers every message within 3 units successfully activates every node once within any interval of 3 steps. By the choice of the message sinks, it follows that the set of successful transmissions in any step forms an independent set, implying that the activation schedule yields a 3-coloring of $G$. It thus follows, that the problem of finding a routing-scheduling pair that minimizes delay is NP-hard. □

Since the proof of the above theorem applies even when the set of routes are specified (one-hop routes, in this case), it follows that it is also NP-hard to determine a delay-optimal activation schedule *given a fixed set of routes*. On the other hand, as we show in Section 4, a set of delay-optimal routes for a given activation schedule can be obtained in polynomial time.

These results indicate that the general problem of designing an optimal data dissemination protocol, given an arbitrary sensor workload, is intractable. In this article, we focus on one element of the design space, namely that of first developing an interference-free schedule for edge activation, and then designing delay- or energy- optimal routes given this schedule.

## 3. OUR MODEL

### 3.1 Partitioning Into Cells

We partition the network into square cells, where the length of each cell is set so that a node anywhere in a cell can typically communicate directly with nodes in any of its four horizontal and vertical neighbor cells. This constrains the cell length $L$ to be at most $R/\sqrt{5}$, where $R$ is the transmission range of a node. In practice, radio irregularities as discussed in Ganesan et al. [2002]; Kotz et al. [2003]; Zhao and Govindan [2003]; Cerpa et al. [2003]; Aguayo et al. [2004] and Zhou et al. [2006] may prevent two nodes in adjacent cells from establishing a high-quality symmetric link. The extent to which our scheduling and routing algorithms can handle radio irregularities is discussed in Section 3.2. Although cells are organized in a rectilinear grid, we make no assumption about the placement of sensor nodes in the grid area. Given an arbitrary node placement, some cells may be empty of nodes.

Each node determines the cell that it belongs to based on its position estimate, which is derived using GPS or distributed localization techniques [Bahl and Padmanabhan 2000; Bulusu et al. 2000; Langendoen and Reijers 2003].

This simple partitioning scheme allows us to schedule communication tasks at the cell level, rather than at the node level. Our proposed wave schedules concurrently activate cells that are sufficiently spaced apart so that the message transmissions within these cells do not interfere with one another.

Distributed localization techniques may result in a node making an inaccurate estimate of its position. Given the cell size $R/\sqrt{5}$, small errors often have no impact, since the node still classifies itself to the correct cell with high probability. Larger and more infrequent localization errors, which misclassify nodes into neighboring cells, simply result in nodes engaging in communication when these cells are activated. Their transmissions may interfere with other transmissions that occur less than 6 cells away. As we discuss in Section 4, our wave schedules concurrently schedule transmissions that are spaced apart by 7 cells thus avoiding interference due to localization errors. Huge localization errors are expected to increase interference, but the effect will be local to the nodes with the wrong position estimates. In all cases, our algorithms continue to operate, but their ability to avoid interference degrades in the presence of very large errors.

### 3.2 Assumptions on the Radio Model

We rely on the assumption that most radios transmit at similar power, and exhibit qualitatively similar radius coverage maps. Realistic radius coverage maps, which show the probability of successful packet reception at each point around a Mica sensor node, are provided by Ganesan et al. [2002]. It is widely accepted [Ganesan et al. 2002; Kotz et al. 2003; Zhao and Govindan 2003; Cerpa et al. 2003; Aguayo et al. 2004] that radio propagation is nonisotropic (connectivity is not the same in all directions from the sender at a given distance) and features nonmonotonic distance decay (lower distance does not mean better link quality) and asymmetrical links. In this article, we adopt the preceding realistic assumptions; in addition, however, we require the following restrictions:

**(A1)** *We assume that up to a certain distance from the sender (the length of the cell's diagonal), packet reception rates are uniformly and consistently high.* To detect if this condition is satisfied during a network setup phase, we could run standard link quality tests used in previous studies of low-power radio models [Ganesan et al. 2002; Zhao and Govindan 2003; Cerpa et al. 2003]. In their recent work, Zhao and Govindan [2003] show that the great majority of nodes located at distance half the communication range from the sender reported reception rates higher than 95%. They also report that the reception rate variance with time is low for all receivers up to a certain distance. Ganesan et al. [2002] further report that *at short distances from the transmitter, a negligible percentage of links are asymmetric*. These studies show that our first constraint is realistic in the particular class of low-power Mica nodes.

**(A2)** *We assume that there exists at least one node per nonempty cell, referred to as a high-fidelity node, that can establish reliable symmetric links with at least one counterpart in each nonempty adjacent cell, and that the network of these high-fidelity nodes remains connected to the data sinks. The network* *should be dense enough so that connectedness is ensured by using grid-like links—links between nodes in adjacent cells—and very few diagonal or long links.* To detect if this condition is satisfied during a network set upphase, we could use a variant of GAF to select a high-fidelity node for each cell (e.g. the node connected reliably with the largest number of neighbors in adjacent cells), and then a flooding variant to detect if the network of high-fidelity nodes is connected to the sinks mostly thru grid-like links.

We address the question of whether the above conditions are satisfiable by real networks, by referring to the experimental analysis of Mica 1 and Mica 2 networks in Cerpa et al. [2003]. They report that the great majority of Mica 2 motes transmitting at medium-high power in an outdoor habitat have very high delivery probabilities at all distances within communication range (and thus among high-fidelity nodes in adjacent cells). Hence, this network would be suitable to run our wave scheduling algorithms. However, Mica 1 motes transmitting at medium power in an indoor office were shown to exhibit great variation in reception rate for almost all the distances tested, making our wave scheduling algorithms unsuitable in this case. To clarify, we do not require that all links in our network are symmetric, high-throughput, and consistently reliable. We only require that there exist a small subset of high-fidelity nodes (at least one per cell) that can establish such high-quality links with their counterparts in the four adjacent cells.

We believe that the algorithms that we discuss in the next section make an interesting contribution that will have impact in practice, however we have not yet studied them in a real deployment. Our experimental results were obtained in a simulator where we compared our algorithms to other approaches. We thus believe that there will be additional systems issues that need to be addressed to make our algorithms truly practical in a real deployment. The focus of this article is on the conceptual ideas; systems issues are the focus of future work.

## 4. WAVE SCHEDULING

The results of Section 2 indicate that the general problem of designing an optimal data dissemination protocol, given an arbitrary sensor workload, is intractable. In this section, we focus on one element of the design space, namely that of developing a schedule for edge activation that avoids interference. We present a class of periodic schedules that are aimed at avoiding collisions at the MAC layer. In Section 5 we present routing protocols that achieve key design goals and describe how to integrate routing and wave scheduling.

Given a set of cells arranged in a rectilinear grid, we propose a class of periodic activation schedules that conserve energy by (i) avoiding interference at the MAC layer and (ii) allowing nodes to turn off their radios whenever they are not sending or receiving messages. In these schedules, which we call *wave schedules*, every directed edge of the rectilinear grid that connects two adjacent cells is activated periodically at well-defined communication intervals, called *send-receive* intervals, of duration $\delta$. For any two adjacent cells $A$ and $B$, the edge $A \to B$ is activated in the send-receive intervals $[t + iP, t + iP + \delta]$, for every $i \geq 0$, where $t$ is the first time the edge is activated, and $P$ is the period of the schedule.

To explain how nodes participate in communication tasks during an edge activation, we need to distinguish between *router* nodes and *leaf* nodes. Router nodes (or *routers*) forward the data of other nodes, as well as their own, towards a sink. Similarly, leaf nodes (or *leaves*) forward only their own data towards a sink. A node may play the role of a leaf with respect to one sink and the role of a router with respect to another; these roles are assigned by the routing protocol. The proposed wave schedules do not assume knowledge of which nodes are leaves and which are routers; they define activation schedules at the cell level, and indirectly determine how nodes will schedule their transmissions once they know their routing roles. We are now in a position to present the edge activation step, and describe in detail the two wave schedules: SimpleWave and PipelinedWave.

**Edge activation**. An edge activation $A \rightarrow B$ consists of two time intervals: a *local* interval reserved for propagating local traffic to routers of cell A, and a *transit* interval reserved for forwarding route-thru traffic to routers of cell $B$.

During the local interval, all routers in cell $A$ turn on their radios in order to receive leaf-to-router traffic. Leaf nodes willing to communicate with these routers turn on their radios for transmission. Notice that we restrict the location of the receiving routers to be in cell $A$,[1] whereas the sending leaves could be located in or around cell $A$ (see Figure 1). The choice of which nodes act as routers in cell $A$, and which neighboring nodes act as leaves, is made at the routing layer. Our proposed routing mechanism in Section 5.3 is designed so that leaves are typically located in the same cell ($A$) as their routers, and there exists one router per cell (as shown in Figure 2). Hence, in most cases, leaf-to-router traffic concerns intracell communication. Contention resolution MAC protocols work very well in avoiding intracell contention, since nodes in the same cell are usually within communication range and there are rare occurences of the hidden terminal problem. Due to radio irregularities, however, it is possible that some of the routers and leaves involved are not fully connected, thus increasing the chance of channel reservation failure. Lack of connectivity is rare considering that the distance of any two nodes in the same cell is much smaller than the transmission range ($dist \leq \sqrt{2}L = \sqrt{2}R/\sqrt{5}$).

Before we present the transit interval, we clarify the notion of direction i) for an *edge* connecting two adjacent cells, and ii) for a wireless *link* connecting two nodes. The direction of an edge $A \rightarrow B$ from cell $A$ to cell $B$ is North, East, South, or West, if cell $B$ is to the north, east, south, or west of cell $A$ respectively. For instance, the direction of $A \rightarrow B$ in Figures 1 to 4 is East. The direction of a link $N_1 \rightarrow N_2$ depends on the relative position of the two nodes. Let $dx = x_2 - x_1$ and $dy = y_2 - y_1$, where $(x_1, y_1)$ and $(x_2, y_2)$ are the coordinates of $N_1$ and $N_2$ respectively. We define the direction of the link $N_1 \rightarrow N_2$ to be:

—North, if $-dy < dx < dy$

—South, if $dy < dx < -dy$

[1]If a router is not in cell $A$, it will be activated to receive local traffic in a different local interval.

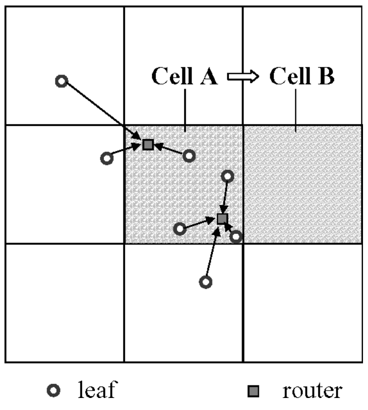

Fig. 1. General case of local interval.

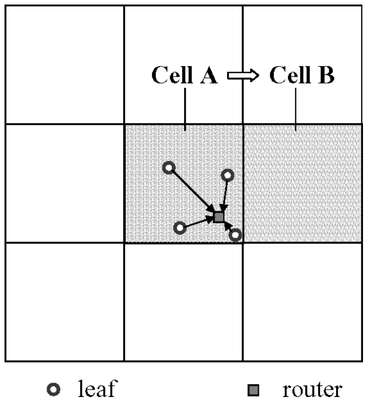

Fig. 2. Common case of local interval.

—East, if $-dx < dy < dx$
—West, if $dx < dy < -dx$

We say that a link between two nodes is east-going if it has the east direction (similarly for the other three directions).

The transit interval of an edge activation $A \rightarrow B$ is reserved for router-to-router traffic sent to cell $B$ *in the direction of edge* $A \rightarrow B$. If cell $B$ is located to the east of cell $A$, routers of cell $B$ turn on their radios to receive router-to-router traffic propagated over east-going links. For example, in Figure 3, messages can be sent along three east-going links, namely $R_5 \rightarrow R_4$, $R_2 \rightarrow R_1$ and $R_3 \rightarrow R_1$. The receiving routers ($R_1$ and $R_4$) must all be in cell $B$. The only restriction for transmitting routers is that they must form east-going links with the receiving routers.

The set of nodes involved in message exchange during the transit interval is determined at the routing layer. Our proposed routing mechanism is designed to typically select a single transmitting router located in cell $A$, which communicates with a single receiving router in cell $B$. Thus, when our routing

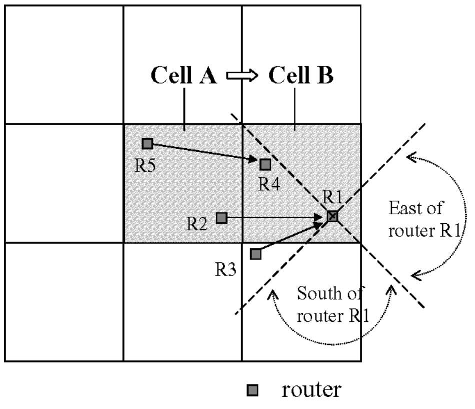

Fig. 3. General case of transit interval.

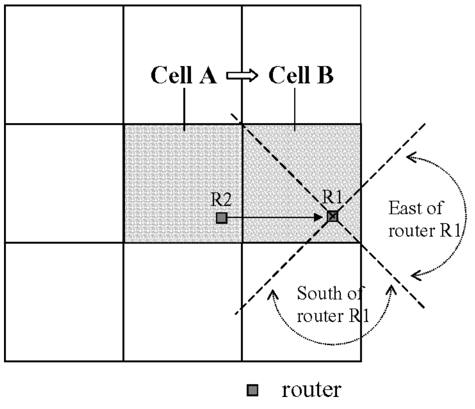

Fig. 4. Common case of transit interval.

mechanism is coupled with the edge activation schedule, the combined scheme usually results in message exchange between a single pair of nodes located in adjacent cells (Figure 4).

If a sending router has no data messages to send, it sends a special *Nothing-ToSend* (NTS) message, which allows both sender and receiver routers to turn off their radios without having to wait until the end of the *send-receive* interval. In case a router receives messages from more than one sending router in a given direction, it must receive NTS messages from all of them before turning off the radio. As we will show in the experimental section, the use of NTS messages offers significant energy savings since it adjusts the node duty cycle to its local traffic.

The main goal of SimpleWave and PipelinedWave is to schedule only noninterfering cell edges concurrently. When only noninterfering cell edges are activated concurrently, and each edge typically involves communication between a single pair of router nodes, there is very little interference in the wireless medium. Hence, it is not necessary to exchange RTS and CTS messages prior to sending a regular data message (or an NTS message). A data (or NTS) message is simply followed by an ACK. The first data or NTS message sent to *B*

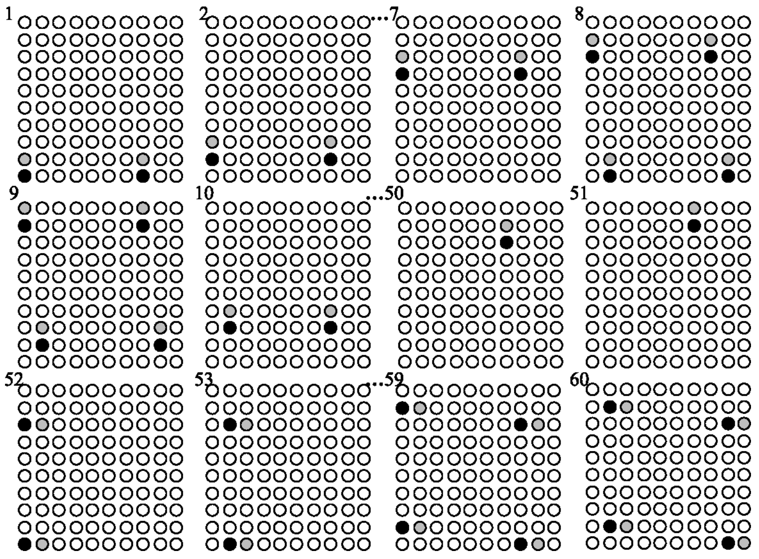

Fig. 5. SimpleWave.

(and its ACK) can be used in order to resynchronize the clocks of the sender and the receiver nodes for the next activation of edge $A \to B$.

In the remainder of this article, by edge activation we mainly refer to the transit interval of the edge activation used for router-to-router communication in a specific direction. The ratio of the transit interval to the local interval depends on the traffic patterns of the application. For instance, for traffic workloads with messages following multiple hops before reaching the destination, the transit interval should dominate the local interval.

**SimpleWave**. The intuition behind wave schedules is to coordinate message propagation in north, east, south and west phases. For instance, during the east phase, only edges of the form $(i, j) \to (i + 1, j)$ are activated sending messages along the east direction. Owing to interference, however, we cannot schedule all of the edges along the east direction. Let $\Delta$ denote the ratio of the interference range to the transmission range, $R$ the transmission range and $L$ the length of the side of a cell. A sufficient condition for activating cells $(i, j)$ and $(i_1, j_1)$ to start receiving messages in a given direction without interference is the following:

$$\sqrt{(i - i_1 - 1)^2 + (j - j_1 - 1)^2} \cdot L \geq \Delta \cdot R.$$

In particular, if we consider two cells $(i, j)$ and $(i_1, j)$ in the same column, then their activations do not interfere if $i - i_1 \geq \Delta R/L + 1$. Since $i - i_1$ is an integer, we obtain that nodes in the two cells can receive simultaneously if they are spaced apart by at least $g$ cells ($i - i_1 \geq g$), where $g = \lceil \Delta \cdot R/L \rceil + 1$. If we adopt the IEEE 802.11 settings of $R = 250m$ and $\Delta = 550/250$, and set $L$ to its maximum value $R/\sqrt{5}$, we obtain that $g = 6$.

In the SimpleWave schedule, we schedule together edges that are $g$ positions apart. Figure 5 illustrates the SimpleWave schedule on a $10 \times 10$ network, with

$R = 250m$, $\Delta = 550/250$, setting $L$ to a round number of $100m$ (instead of its maximum value $R/\sqrt{5}$), yielding $g = 7$. The higher value of $g$ ensures that small localization or synchronization errors will not cause interference. The north phase starts at time 1 and it lasts for 51 send-receive intervals during which every north edge is activated exactly once. The following east phase starts at time 52. Notice that only two cells of the second column ($(1, 0)$ and $(1, 7)$) which are spaced apart by 7 hops, are receiving concurrently in the east direction. In the next interval (time 53) the pattern shifts east by one cell. Only when the wave has propagated to the eighth and ninth columns (time 59) does it no longer interfere with node communication in the first two columns. Note that at time 59 it becomes possible to concurrently schedule four edges: $(7, 0) \rightarrow (8, 0)$, $(7, 7) \rightarrow (8, 7)$, $(0, 1) \rightarrow (1, 1)$ and $(0, 8) \rightarrow (1, 8)$.

There are variants of this *SimpleWave* algorithm, differing by the order in which wave directions are scheduled. We refer to these as the $(N, E, S, W)$, $(N, W, S, E)$, $(N, S, E, W)$, and so forth. The variants are logically equivalent, but the choice of scheduling variant affects the choice of routes, as will be shown in Section 5.3. The period of a *SimpleWave* depends on the size of the network. Each phase takes $(N - 1) + (g - 1)g$ send-receive intervals and the entire wave period lasts for $4((N - 1) + (g - 1)g)$ intervals. This is not a desirable property, because it prevents the distributed deployment of the algorithm in a dynamic network. When a new node (in a new cell) joins (or leaves) the network, it affects the wave period and therefore the activation times of all the other cells. In addition, in order to identify its activation schedule, the new node must not only know its location in the network, but also the size of the network. Another important downside of the *SimpleWave* algorithm is that it underutilizes the capacity of the network. For instance notice in Figure 5 that at time 1, it activates only two edges, whereas one could identify two additional edges that could be activated concurrently without causing any interference.

**PipelinedWave**. This algorithm is motivated by the need for schedules that can be deployed in a distributed and scalable manner, and that make good use of the network capacity. Conceptually, a network can be divided into a number of fixed-size ($g \times g$) areas of $g^2$ cells each, which we refer to as *squares*. For example, the network in Figure 6 has 9 squares and each square consists of 49 cells. Since all edges within the same square interfere with one another, we can only schedule one edge at a time.

In the simplest version of PipelinedWave, we set all squares to have identical schedules and, in each square, the schedule of the incident edges of a cell is determined by its relative location in the square (Figure 6). In effect, we partition all the edges of the network into a collection of *maximal independent sets*, each independent set corresponding to a set of edges that can be simultaneously activated without interference. A maximal independent set includes one edge per fixed-size square, and all such edges are located in the same position within each square. The period of the resultant schedule is $4g^2$ send-receive intervals, a sufficient number of intervals to activate all edges within a square. This means that for pipelined waves, new nodes can join the network and schedule themselves without affecting the schedules of existing nodes. If a node is in a cell of an existing square, it waits for at most one period in order to interact

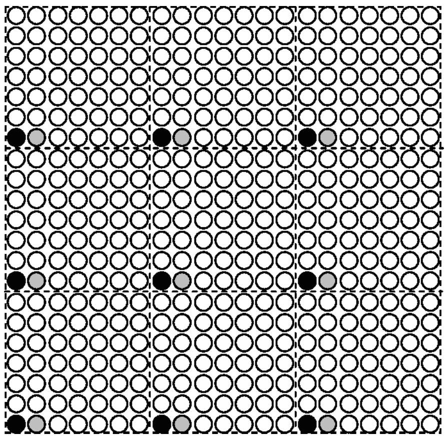
Fig. 6. Simple version of PipelinedWave in which squares have identical schedules.

with its neighbors and locally determine its location with respect to them and therefore its local coordinates within the square. By overhearing the schedules of its immediate neighbors it determines the time at which it should schedule itself in each direction. A similar local interaction occurs when a node joins the network initializing a new cell in a new square. When a node leaves the network, the schedules of the remaining cells and their nodes do not change.

Note that in the *PipelinedWave* algorithm, two edges are scheduled concurrently if they have the same direction and relative position within a $g \times g$ square. Thus the algorithm significantly reduces interference at the MAC layer. It schedules a maximum number of noninterfering edges at each send-receive interval thus increasing the network capacity with respect to the *SimpleWave* algorithm. It is easily deployable in a distributed manner, since local coordination suffices for scheduling a new node. Finally, it is scalable because the node schedules are not affected by the size of the network.

A modified version of the *PipelinedWave* algorithm does not define identical schedules for each square, but schedules shifted by $g$ positions with respect to the schedules of the four neighbor squares (as shown in Figure 7). More specifically, the east wave of a square is shifted $g$ positions (send-receive intervals) earlier than the east wave of the west neighbor square, the north wave is shifted $g$ positions earlier than the north wave of the south neighbor square and so on. The east phase in a given (dotted) square proceeds by shifting one edge to the right and moving to the row below when the entire row of the square is traversed. Notice that by the time an entire row is traversed in a given square, the respective row of the right neighbor square just starts being traversed. The new pipelined algorithm decreases the delay of message delivery at the square boundaries, as will become evident when we describe *delay-based* routing in Section 5.3. The south, west and north phases are scheduled in a similar manner. This improved *PipelinedWave* is the schedule evaluated in our experiments in Section 6.

Another tunable parameter in *PipelinedWave* is the number of send-receive intervals for each direction (phase) before the wave switches to another

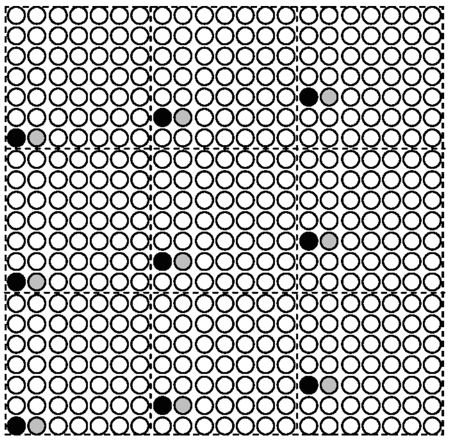

Fig. 7. PipelinedWave in which squares have shifted schedules.

direction. Our experiments show that this parameter, referred to as *step*, has no noticeable impact on the performance of the wave schedule (Section 6).

**Synchronization**. We briefly discuss two synchronization requirements imposed by wave schedules: i) neighboring nodes must have the same notion of time regarding their communication slot and ii) nodes in the *close* neighborhood must be well synchronized so that only edges at least $g$ positions away are scheduled simultaneously. Since perfect time synchronization is hard to achieve, we relax the initial requirements and propose a *fault-tolerant* version of wave schedules. If the drift between two neighbor clocks does not exceed $\epsilon$, nodes that are $g$ positions away from each other are synchronized within $g\epsilon$. In every edge activation, we schedule the receiver to turn on the radio $\epsilon$ time units earlier than the scheduled time according to its local clock. In order to ensure that there is going to be no interference due the clock errors, we can increase the distance between two noninterfering edge activations (e.g. from 7 to 8). Recently proposed synchronization protocols for sensor networks (e.g., RFA [Werner-Allen et al. 2005], RBS [Elson et al. 2002] and TPSN [Ganeriwal et al. 2003]) provide tight synchronization bounds (in the order of $\mu secs$) and exhibit good multi-hop behavior. Their performance however is bound to decay for very large networks; in this case we assume that a few GPS-equipped nodes will undertake the synchronization task for their local regions.

## 5. ROUTING

In this section, we discuss how to integrate routing protocols with a wave schedule. Wave schedules are general purpose TDMA-based MAC protocols that can be combined with arbitrary routing protocols that use unicast communication. An arbitrary routing protocol, however, may not fully exploit the collision avoidance properties of wave schedules. We first identify the potential causes of interference, and present the properties that a routing algorithm must have in order to minimize the probability of collisions.

## 5.1 Routing for Wave Scheduling

A wave schedule defines a periodic schedule of activating edges between two adjacent cells. A routing protocol determines the routes in which messages are forwarded to a sink. To combine routing and scheduling, we need to specify how to schedule a 1-hop message transmission from node $N_{snd}$ to $N_{rcv}$. Note than $N_{snd}$ could be either a leaf or a router, whereas $N_{rcv}$ is always a router (with respect to the sink where the message is destined). We first identify the cell $C$ in which $N_{rcv}$ is located. Let $C_N$, $C_E$, $C_S$, and $C_W$ be the cells adjacent to $C$ in the north, east, south, and west directions respectively.

—If $N_{snd}$ is a leaf, the transmission is scheduled in the first available local interval of any of the four edge activations: $C \rightarrow C_N$, $C \rightarrow C_E$, $C \rightarrow C_S$, or $C \rightarrow C_W$.

—If $N_{snd}$ is a router, the scheduling of the message transmission depends on the direction of $N_{snd}$ with respect to $N_{rcv}$ . If $N_{snd}$ is located in the west of $N_{rcv}$, the transmission is scheduled in the first available transit interval of edge $C_W \rightarrow C$. Note that $N_{snd}$ is not necessarily located in cell $C_W$ (e.g. observe the location of router $R_3$ in Figure 3). Similar rules hold for the remaining three directions.

This means that our wave algorithms are designed to schedule any unicast transmission selected by the routing protocol, regardless of the network density or the irregularities of the radio model (e.g. asymmetrical links, nonmonotonic distance decay, nonisotropic connectivity). The question is whether and under what conditions wave-scheduled transmissions are collision-free.

## 5.2 When Are Wave-Scheduled Transmissions Collision-Free?

Combining an arbitrary routing protocol with a wave schedule does not necessarily guarantee collision-free message transmissions. In order to minimize the probability of collisions, a routing protocol needs to satisfy the following properties:

—**Property 1:** In local intervals, the condition to avoid interference during local intervals is to have *all leaf nodes in a cell send their local data to a single router node located in the same cell*, as shown in Figure 2. This condition is satisfied under our assumption (A1) of Section 3.2 since nodes within each cell will be able to reserve the medium and transmit without interference during the local intervals of the wave.

—**Property 2:** In transit intervals, the condition to avoid interference is to have *a single pair of router nodes communicating route-thru traffic per edge activation*, as shown in Figure 4. This condition holds under our assumption (A2) of Section 3.2 about the connectivity of high-fidelity nodes across cells.

If the transmissions selected by a routing protocol satisfy the aforementioned properties in most cells, we say that the routing protocol exhibits a *wave-friendly* behavior. In the next section, we propose a routing protocol that is especially designed to exhibit wave-friendly behavior in forwarding data from source sensor nodes to a few gateway nodes.

## 5.3 Routing Protocol

The routing protocol that we propose in this section is designed to exploit the potential of wave schedules to reduce interference, while remaining robust to irregularities of the radio model (anisotropy, asymmetry, non-monotonic distance decay). The suitability of derived routes to wave schedules depends on the intensity of radio irregularities. In an ideal radio model, the combined routing and scheduling protocols eliminate interference at the MAC layer, whereas their performance degrades gracefully as we move to more adverse radio propagation conditions.

**Step 1: Initial router selection.** A node initially identifies as neighbors, the nodes with which it can establish reliable symmetric communication links. Each node broadcasts its location and residual energy to its neighbors. The node that has the largest residual energy among the neighbors located in the same cell assumes the role of a router for every sink. Every other node assumes the role of a leaf for every sink. The set of routers selected in this step is not final, since more routers may be added in Step 3. The initial step of router election is similar to the leader election mechanism in GAF [Xu et al. 2001]. Note that radio irregularities may result in electing more than one router per cell. Such cases are rare considering that the maximum distance of two nodes in the same cell is much smaller than the radio transmission range.

**Step 2: Path selection.** Flooding of routing messages is initiated at each sink $S$, in order to establish paths from source nodes to $S$. Consider a path $N \rightarrow R_1 \rightarrow \ldots \rightarrow R_n \rightarrow S$ that connects a node $N$ to sink $S$. One of the design goals of a routing algorithm (discussed in Section 5) is that every link $R_i \rightarrow R_{i+1}$ connecting two routers must cross only one cell boundary, which implies that $R_i$ and $R_{i+1}$ are located in adjacent cells. We refer to such links as *regular* links. All remaining links, e.g. links connecting two routers in nonadjacent cells, or links connecting a leaf and a router are referred to as *irregular* links.

The first goal of flooding routing messages is to identify paths with a minimum number of irregular links from source nodes to sinks. Each node stores locally the number of irregular links to each sink initially set to infinity. A routing message originated at a sink includes the sink identifier $S$ and the number of irregular links $IL$ to the sink, initially set to 0. When a node $N$ receives a routing message from a neighbor $N'$, it increments the number of irregular links in the message ($IL = IL + 1$) if and only if $N$ and $N'$ form an irregular link. Then it compares the new $IL$ with the locally stored number of irregular links for sink $S$, denoted as $Min\_IL$.

(1) If $IL > Min\_IL$, then the routing message is discarded, since it offers a path from node $N$ to sink $S$ with more irregular links than the best path known so far.

(2) If $IL < Min\_IL$, then the information in the routing message is adopted as the best-known routing option from node $N$ to sink $S$, by setting $Min\_IL := IL$. This means that, unless better routing information is received, $N$ will select neighbor $N'$ to forward all messages destined to sink $S$.

(3) If $IL = Min\_IL$, then the decision depends on the routing metric used to evaluate the efficiency of a routing algorithm. We consider two metrics, namely *node energy consumption* and *message propagation delay*—referred to as the minimum energy and the minimum delay metric. In what follows, we discuss the impact of each metric on route selection.

In order to explain how the energy and delay metrics are used, we introduce a few more fields to be included to be in routing messages. Besides the number of irregular links, a routing message broadcast by node $N'$ includes i) the number of regular links ($RL$) from $N'$ to sink $S$, and ii) four delay-related fields ($D_{North}$, $D_{East}$, $D_{South}$, $D_{West}$). The values of the delay-related fields strongly depend on the wave schedule. The field $D_{North}$ stores the delay between the time that $N'$ receives a message on a north-going link (i.e. from a neighbor located in the south of $N'$) and the time that the message is delivered to the sink.

On receiving a routing message from neighbor $N'$, node $N$ increments the number of regular links ($RL := RL + 1$) if and only if $N$ and $N'$ form a regular link. Node $N$ also updates the delay-related fields to reflect the delays of message propagation from $N$ to sink $S$. For example, if $N$ were to forward a message through node $N'$, and node $N'$ is in the east direction of $N$, the delay fields of the message would be updated as follows:

—$D_{North} := WaitingDelay(N, North, N', East) + D_{East}$
—$D_{East} := WaitingDelay(N, East, N', East) + D_{East}$
—$D_{South} := WaitingDelay(N, South, N', East) + D_{East}$
—$D_{West} := WaitingDelay(N, West, N', East) + D_{East}$

where $WaitingDelay(N, dir_1, N', dir_2)$ denotes the time that a message waits in the queue of node $N$ from the time that it is received by $N$ in the direction $dir_1$ to the time that it is sent from $N$ to $N'$ in direction $dir_2$.

We now consider how the updated fields of the routing message impact the behavior of the routing algorithm. If the energy metric is used, the current node $N$ adopts the message routing information, only if it offers a smaller number of regular links than the best route so far. If the delay metric is used, the current node $N$ tries to minimize the total delay from the time that it receives a message from a certain direction to the time that the message is delivered to the sink. Node $N$ compares the updated delay fields of the message ($D_{North}$, $D_{East}$, $D_{South}$ and $D_{West}$) with the best-known routing delays so far, which are derived from previous messages. We denote the latter as $Min\_Del_{North}$, $Min\_Del_{East}$, $Min\_Del_{South}$, $Min\_Del_{West}$. If $D_{North} < Min\_Del_{North}$, then node $N$ will adopt the updated message information setting $Min\_Del_{North} := D_{North}$; this means that the node will forward all messages received in the north direction to node $N'$, expecting to minimize the total propagation delay. A different decision may be made for messages received by $N$ in another direction, e.g. if $Min\_Del_{South} < D_{South}$, node $N$ does not select $N'$ to forward messages that it receives in the south direction.

**Step 3: Final router selection:** When this algorithm converges, some of the nodes that were elected to be leaves in Step 1, are upgraded to routers. In particular, if a leaf node $N$ is set to receive messages from a neighbor node in order to forward them hop-by-hop to sink $S$, then the node $N$ becomes a router with respect to sink $S$. Recall that a node may be a leaf for traffic destined to one sink and a router for traffic destined to another sink. This means that it will propagate messages concerning the first sink during local intervals, and messages concerning the second sink during transit intervals of the wave schedule.

## 6. EXPERIMENTAL EVALUATION

We implemented a prototype of wave scheduling in the NS-2 Network Simulator [Breslau et al. 2000] and compared its performance with other approaches. In Section 6.1, we test the behaviour of wave schedules under different routing metrics, as well as varying the number of sinks, empty cells, nodes and links in the network. We also compare different interleaved wave schedules by varying the number of steps. Section 6.2 presents the performance of two competing tree-based scheduling approaches and Section 6.3 shows the behavior of IEEE 802.11 with various duty cycles. A comparison of wave schedules with the other approaches is presented in Section 6.4. Finally, in Section 6.5, we evaluate the performance of wave schedules using real datasets, including workloads with hotspot traffic.

### 6.1 Wave Scheduling

We simulate a network of 10 by 10 grid cells of size 100m$^2$ each. The ratio of interference to communication range is 550/250 and the ratios between radio idle, receive, and transmit power are 1:1.2:1.6. Whereas in our previous work [Trigoni et al. 2004], we used the two-ray ground reflection model, which represents communication range as an ideal circle, we now use the shadowing model, in which nodes can only probabilistically communicate when near the edge of the communication range, due to fading effects. Nodes initially exchange heartbeat messages, and each node selects as its neighbors, the nodes with which it can establish a reliable symmetric link with very high probability ($> 98\%$).

In our wave schedules, every edge activation between two consecutive cells lasts for 200ms. A node can send about 10 packets during an edge activation given a link bandwidth of 20kbps. The receiver wakes up 30ms before the sender to avoid message loss when clocks are subject to small drifts. The size of a square in a pipelined wave is set to 8 by 8 grid cells (instead of 7 by 7) in order to avoid interference as a result of small localization or synchronization errors. The parameter values of the experimental setup are summarized in Table 8.

**Initial node placement and traffic workload**. Experiments run for 1000 seconds and the traffic workload varies from 0 to 2000 messages. We distribute 350 sensor nodes in the network area uniformly at random. In experiments with empty cells, we first randomly select a set of empty cells, and then we distribute nodes in the remaining network area. In our initial experiments we test the behavior of wave schedules using random workloads. That is, we select uniformly at random 1) the sinks over the set of sensor nodes, 2) the time that a message is generated over the simulation period, 3) the node where a message

| parameter | value |
|---|---|
| comm. range | 250 meters |
| interf. range | 550 meters |
| cell width | 100 meters |
| grid size | (10 by 10) cells |
| square size | (8 by 8) cells |
| idle:rcv:trans power | 1:1.2:1.6 |
| send/rcv interval | 200 ms |
| experim. duration | 1000 secs |
| bandwidth | 20kbps |

Fig. 8. Experimental setup.

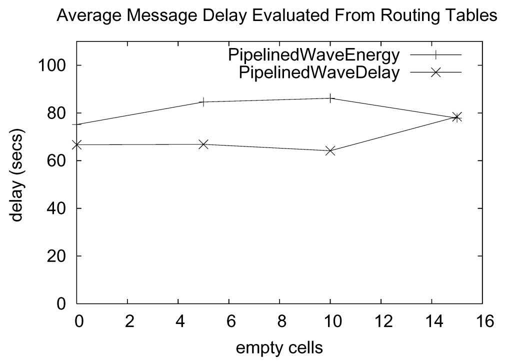

Fig. 9. Delay vs. energy routing.

is generated over the set of sensor nodes, and 4) the final message destination over the set of sink nodes. In Section 6.5, we also test our algorithms with traffic workloads derived from real datasets.

**Energy- vs. delay-based routing**. We first compare the behavior of the Pipelined-Wave schedule under two wave routing metrics: minimum energy and minimum delay. Recall from Sections 4 and 5 that due to the scheduling of the waves, the path with minimum delay is not necessarily the path of minimum hop count. Figure 9 shows the average path delay under light load for the two metrics: it shows the time between generation of a message at its source and delivery of the message at its destination. This delay is computed by deriving information from the routing tables of the nodes. It coincides with the real message propagation delay when the traffic is low and nodes can completely drain their buffer during an edge activation. The minimum energy routing metric defines paths with higher delay than the minimum delay metric. The gap initially increases as we increase the number of empty cells, because the paths selected with the minimum energy metric have fewer regular hops, but they make higher-delay turns to go around empty cells. As we further increase the number of empty cells, the two routing mechanisms converge; the reason is that they make the same routing decisions due to the limited range of routing options from sources to sinks. The energy overhead of this minimum delay metric was observed to be negligible. In the remainder of this section, we use minimum delay as the default routing metric.

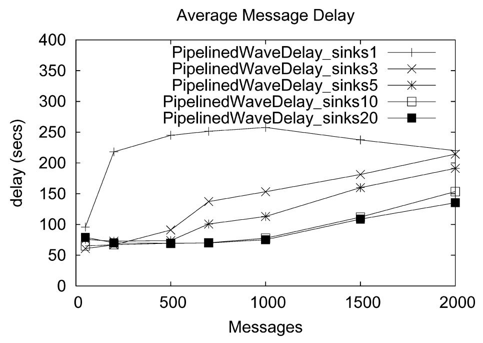

Fig. 10. Effect of sinks on delay.

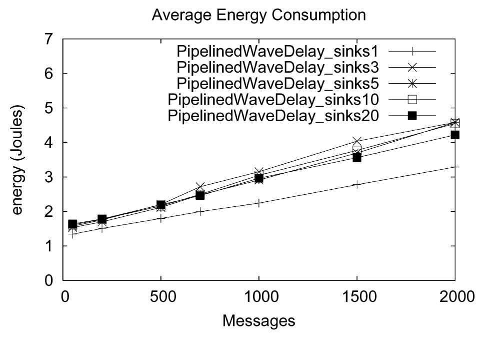

Fig. 11. Effect of sinks on energy.

**Scalability with the number of sinks**. Our second experiment shows the scalability of our scheme with respect to the number of sinks. Figure 10 shows the average observed message delay, which captures queueing delay due to traffic. We set the number of empty cells to be 0. With more sinks, the load is better balanced across the network, the average message propagation delay is smaller, and the overall capacity of the network increases. With more than 50 messages for a single sink, the network is overloaded, the queues in the network start to grow, and they would continue to grow without bounds if we would not have limited the length of the experiment to 1000 seconds. Figure 11 shows that the energy usage of the wave does not increase with the number of sinks, for a given number of messages. The energy with 1 sink is only noticeably smaller because the network is overloaded and most messages stay in queues instead of being propagated. This confirms the nice behavior of wave routing, which makes it exceptionally suitable for sensor networks with multiple sinks.

**Effect of empty cells**. We now examine the impact of empty cells (or *holes*) on the performance of wave schedules. The number of sinks is 10 and a randomly selected set of 0 to 15 cells (out of 100 cells) are set to be empty of nodes. The 350 nodes are distributed uniformly at random in the remaining nonempty cells. Figure 12 shows that the message delay increases with the number of holes: messages wait longer in order to make a turn to bypass a hole. The capacity of the network is only 500 messages with 15 holes (the message delay

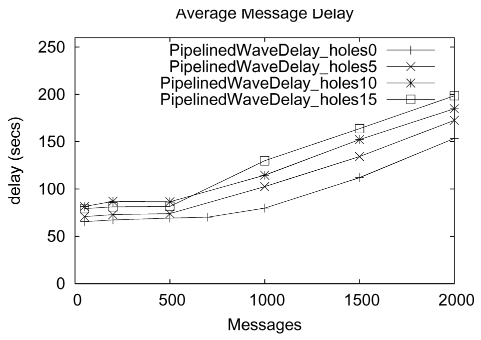

Fig. 12. Effect of holes on delay.

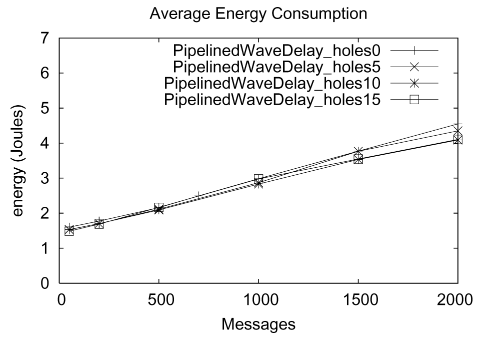

Fig. 13. Effect of holes on energy.

increases considerably after that point), whereas it rises to more than 1000 for networks without holes. The average energy consumption per node is not impacted significantly by the number of empty cells, as shown in Figure 13. This is due to two counterbalancing effects; as we increase the number of holes i) fewer messages are delivered per time unit, thus decreasing the average energy consumption per node, but ii) messages follow longer paths to get to the sinks, thus increasing the average energy consumption per node.

**Effect of asymmetric or nonreliable links:** In all of our experiments we use the shadowing radio propagation model, which omits the unit disk graph assumption, and instead assumes that two nodes within communication range can communicate with a certain probability. The shadowing model evaluates the mean receive power of the radio signal at distance $d$, and its variance, and uses them to model radio propagation. Although this is a more realistic model than the free-space model and the two-ray model, we also tested our algorithms in more adverse network conditions with a significant percentage of low-quality and asymmetric links. In Figures 14 and 15, we show the effect of disabling up to 80% of links of length $len$, where $commRange/2 \leq len \leq commRange$. Notice that as we increase the number of disabled links, the connectivity decreases and messages are routed through longer and higher-delay paths. However, the increase in average message delay and average node consumption is not significant after disabling up to 60% of the links. This shows the robustness of

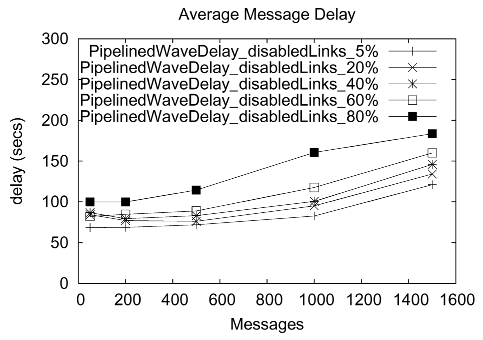

Fig. 14. Effect of removing x% of links that are longer than half the communication range on delay.

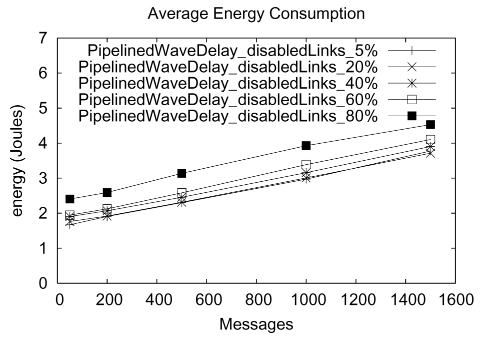

Fig. 15. Effect of removing x% of links that are longer than half the communication range on energy.

our routing and wave scheduling algorithms in the presence of asymmetric or low-quality links, and the graceful degradation of their performance as network connectivity decreases.

Figures 16 and 17 show the performance of our algorithms in a more pessimistic scenario, in which we disable links of all sizes (not only those longer than $commRange/2$). In this case we observe that the performance degrades faster, and both the average message delay and the average node consumption increase significantly after removing more than 10% links. However, even in these adverse conditions, the network continues to operate, delivering messages along fewer and longer paths.

**Effect of network density**. In this experiment, we test the performance of our wave scheduling and routing algorithms as we vary the number of nodes, which we deploy uniformly at random in the network area. Recall that the network area is $1000 \times 1000$ m$^2$ and it consists of $10 \times 10$ cells of size 100m. Figures 18 and 19 show that as we increase the number of nodes, both the average delay per message and the average energy consumption per node decrease. The reasons are that i) all cells gradually become occupied by at least one node, and messages travel from sources to sinks on shorter and faster paths, and ii) the workload is balanced across more nodes, thus reducing the queueing delay and the energy consumption at each node.

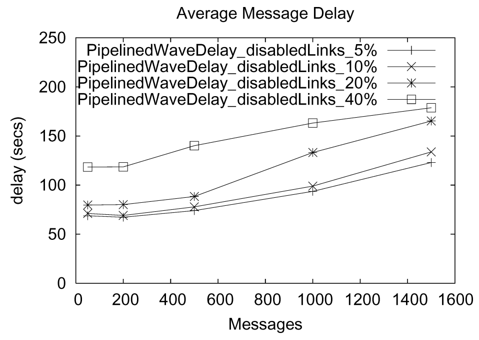

Fig. 16. Effect of removing x% of all links on delay.

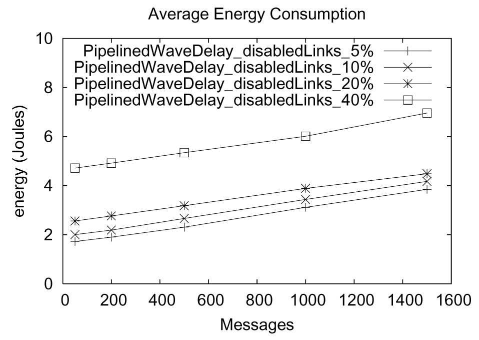

Fig. 17. Effect of removing x% of all links on energy.

**Effect of steps**. Finally, we vary the number of *steps* (from 1 to 16) that the pipelined wave spends in each phase, resulting in different interleaved schedules. We measured the message delay and node energy consumption for waves of 1, 4, 8 and 16 steps, in a network with 10 sinks and 0 empty cells. Since different interleaved schedules select different delay-optimal routes, our expectation was that they would also perform differently in terms of node energy and message delay, especially for networks with many holes. Our experiments, however, showed almost no difference. As shown in Figures 20 and 21 the energy consumption per node and the propagation delay per message are surprisingly similar for different interleaved schedules.

## 6.2 Tree Scheduling

We compare wave scheduling with an existing tree-based scheduling and routing scheme [Madden et al. 2002]. Trees are generated using a flooding mechanism initiated at each sink. Every node selects as its parent the neighbor on the shortest path to the root (sink). It is therefore expected that the paths used in tree schedules are shorter than paths used in waves, since our routing protocol attempts to build the latter on top of the grid overlay. Routing in a tree is trivial: each nonsink node forwards every message it receives to its parent. In a tree-based schedule, we activate edges in reverse order of their distance from the root, enabling a message to propagate from any leaf of the tree to the

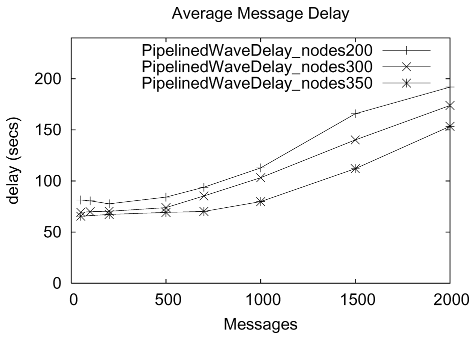

Fig. 18. Effect of node cardinality on delay.

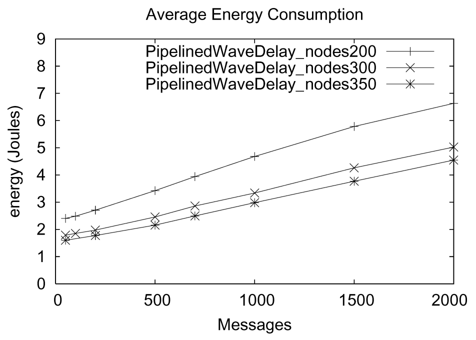

Fig. 19. Effect of node cardinality on energy.

sink in a single tree activation period. Every tree edge is activated for 200ms, as in the case of the wave.

To generalize tree scheduling to handle multiple sinks, we construct a collection of spanning trees, one tree rooted at each sink. An edge activation schedule can then be derived in several ways. At one extreme is a *conservative* schedule, which is simply a concatenation of schedules for the individual trees. The simplest conservative schedule is to activate a tree rooted at sink $i+1$ immediately after all edges of a tree rooted at sink $i$ have been activated. In this conservative schedule, delay grows linearly with the number of sinks. In our experiments we study energy-efficient variants of this simple schedule: We define a period $p$ of repeating the activation of every tree. If we have $m$ sinks, the first tree is activated at times $\{0, p, \ldots\}$, the second at $\{p/m, p + p/m, \ldots\}$, and so on. The interval $p/m$ is long enough to activate all edges of a single tree, so that consecutive activations do not overlap. In Figures 22 and 23, these schedules are referred to as $Tag\_Consec\_Every\_p$, where $p$ is the period between two activations of the same tree.

At the other extreme, we consider *aggressive* schedules that activate all trees in parallel. In the simplest aggressive schedule, which is called $Tag\_Parall$, consecutive activations of the same tree follow one another immediately after completion. In order to study power-saving variants of the aggressive schedules, we consider periodic activations of the same tree. In our experiments, we use

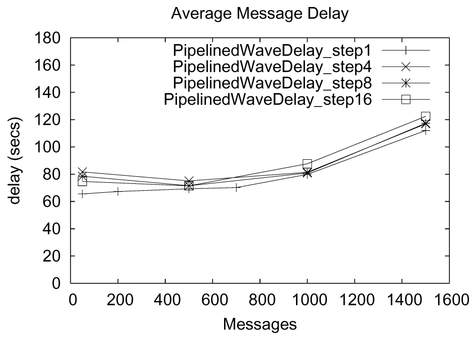

Fig. 20. Effect of steps on delay.

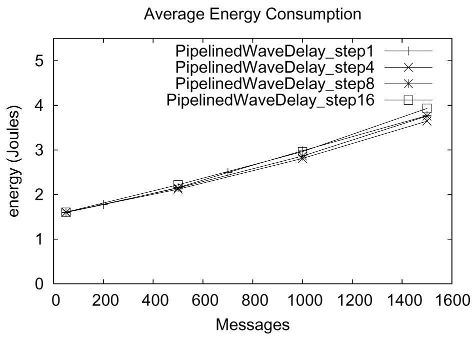

Fig. 21. Effect of steps on energy.

the name $Tag\_Parall\_Every\_p$ to refer to aggressive schedules in which all trees are activated concurrently every $p$ seconds (Figures 24 and 25).

In both consecutive and parallel schedules, we observe a graceful tradeoff between energy and delay. As the activation period increases, the energy decreases at the expense of higher message delay and smaller network capacity. Applications aiming at energy preservation should take into consideration the traffic load in order to determine an energy-efficient tree schedule. For instance, the most energy-efficient consecutive schedule that achieves a capacity of 1500 messages (with delay smaller than 100 seconds) has a period of 50 seconds (Figure 22). Likewise, the most energy-efficient parallel schedule that achieves a capacity of 1500 messages (with delay smaller than 100 seconds) is activated approximately every 8 seconds (Figure 24). Beyond 1500 messages per 1000 seconds, the delay for these two schedules starts increasing rapidly, and it would increase without bounds had we continued to generate messages with the same rate for longer periods.

## 6.3 IEEE 802.11 with Different Duty Cycles

Besides tree scheduling, in which edges are activated in reverse order of their distance to the root, we also study power-conserving variants of the IEEE 802.11 protocol. We vary the duty cycle of the protocol, by turning the radio off regularly and allowing communication only 1 to 20% of the time. The performance of

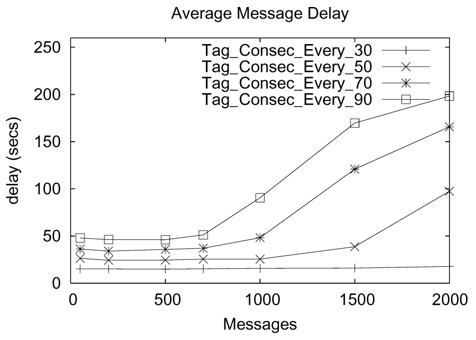

Fig. 22. Delay: consecutive trees.

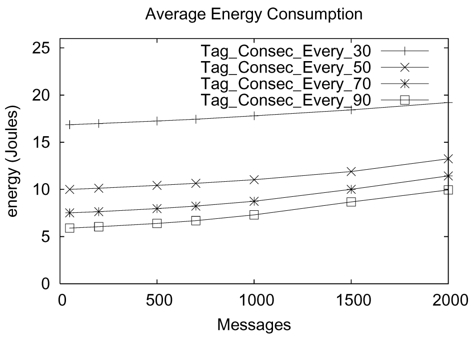

Fig. 23. Energy: consecutive trees.

the resulting schemes, named *DutyCycle_x*, is shown in Figures 26 and 27. Routing is performed as in tree-scheduling: messages follow the shortest paths to the sinks. Notice that for a load of 1500 messages we can only select duty cycles greater than 10%, otherwise the traffic exceeds network capacity and the queues increase without bound. The reader can see trends in energy and delay similar to those observed in the tree-scheduling schemes. As the duty cycle increases, the average message delay decreases significantly at the expense of higher energy usage.

## 6.4 Comparison with Other Schemes

In order to compare different protocols, we first select a traffic load and then consider only protocols that can serve this load without exceeding capacity (the point at which average delay begins to increase rapidly and exceeds 100 seconds). We compare the most energy-efficient versions of different protocols (with 10 sinks and 0% empty cells): for 1500 messages, we select the variants *Tag_Consec_Every_*50, *Tag_Parall_Every_*6, *Duty_Cycle_*10 and the pipelined wave (combined with delay-based routing). From the previous graphs, the reader can see that these are the variants of different protocols that accomodate the given traffic with the least energy consumption.

Figure 28 shows that the wave protocol has the longest delay, followed by the tree-based schedules and the 802.11 (with duty cycle 8%). The reverse pattern

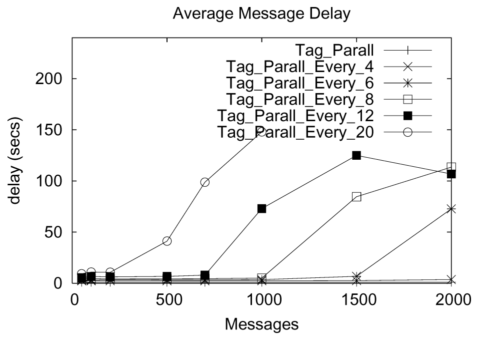

Fig. 24. Delay: parallel trees.

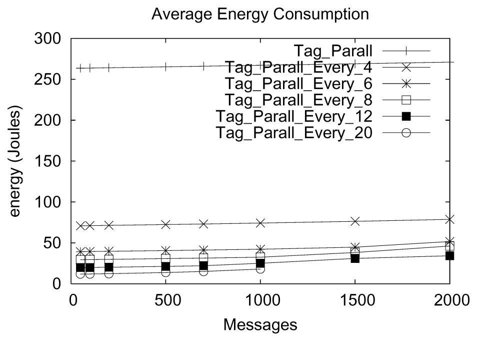

Fig. 25. Energy: parallel trees.

is observed with respect to node energy consumption in Figure 29. The wave protocol is at one extreme, offering the most energy savings (better by an order of magnitude than any other scheme) at the cost of higher delay. The 802.11 protocol with duty cycle 8% is at the other extreme offering very small message delays at the cost of higher energy. The energy-delay tradeoff of the two tree scheduling algorithms is also worth observing: activating trees consecutively (as opposed to concurrently) saves energy because it avoids interference among different trees, but it incurs higher message latencies.

## 6.5 Experiments with Real Datasets

This section evaluates our scheduling and routing algorithms using three real workloads of GPS data. The first dataset contains 24 paragliding flights recorded by a pilot in the Appennini mountains, in Italy. The paragliding dataset has 38,312 (longitude, latitude, time) entries. The second dataset includes the tracks of 137 tourists through the Island of Madeira. The tracks are not free, but heavily constrained by the paths already present on the island. The tourist dataset has 18,776 entries. The third dataset includes the tracks of 8 visitors to the London zoo, amounting to a total of 1857 entries.

We scaled the size of the area in which the objects move in each dataset to the default size of our network, $1000 \times 1000$ m$^2$, and the time interval in which they move, to the default simulation time of 1000 seconds. When a moving object

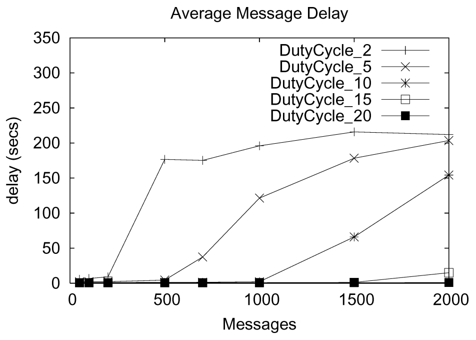

Fig. 26. Delay: 802.11.

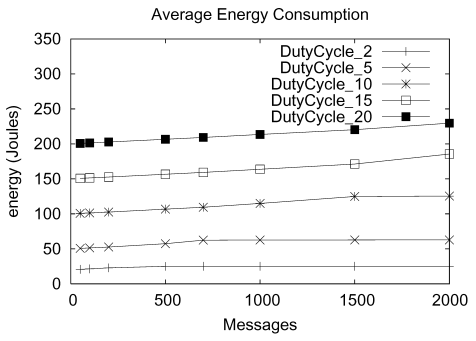

Fig. 27. Energy: 802.11.

passes from location $(x, y)$ at time $t$, we generate a message at the sensor node located closest to $(x, y)$. Hence, the time and the source node where a message is generated, are determined by the real workloads. All messages corresponding to the same moving object have the same final destination, which is randomly selected as one of the 10 available sinks. Sinks are selected randomly over the set of sensor nodes as in the previous experiments.

Figures 30 and 31 show the performance of our wave scheduling and routing algorithms using the three real workloads of GPS data. We chose to process two data workloads from each dataset, one with 200 GPS entries and one with 500 GPS entries. The tourist and zoo datasets yield very similar message delays to the random traffic workload (see lineplot with 10 sinks in Figure 10), whereas message propagation is slower in the paragliding dataset. The three datasets yield very similar average energy consumption per node for both workloads. The energy spent per node is almost the same as in the case of random traffic workloads (see lineplot with 10 sinks in Figure 11).

Finally, we simulated a scenario in which GPS readings are generated in bursts during small intervals, or in small parts of the network. We extract 200 GPS readings from the zoo dataset, and scale them differently in terms of time and space, resulting in different traffic workloads. Although we use the same network area ($1000 \times 1000$ m$^2$ occupied by $10 \times 10$ cells) and simulation time (1000 secs), we select different *active areas* and *active intervals* to 'project'

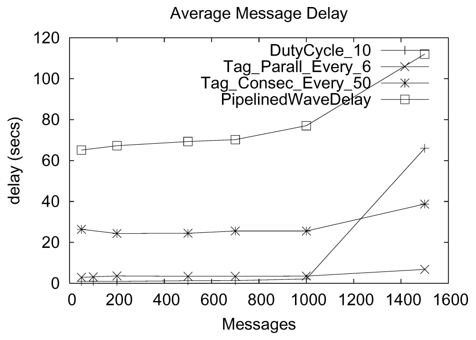

Fig. 28. Comparing schemes.

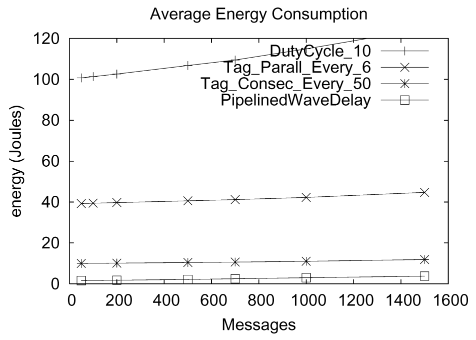

Fig. 29. Comparing schemes.

the tracks of moving objects. Figure 32 shows delay measurements for active areas of 9 and 100 cells, and active intervals of 100 and 1000 secs. First, we observe that messages take longer to be propagated when they are generated in bursts in short time intervals (columns 1 and 3). We also observe no significant change in the message delay when we reduce the active area in which messages are generated (columns 1 and 2), unless the load has exceeded the network's capacity (columns 3 and 4). Figure 33 shows that the energy spent at each node primarily depends on the total number of messages propagated across the network and is hardly impacted by the time interval and area in which messages are generated.

## 7. RELATED WORK

**MAC protocols:** IEEE 802.11 [Society 1999] is the most widely used contention-based protocol; although nodes can periodically switch to a power saving mode, in the active periods they suffer from interference and overhearing. The PAMAS MAC-level protocol turns radios off when nodes are not communicating [Singh et al. 1998], but it requires a second channel for RTS-CTS messages. PicoNet also allows nodes to turn off their radios [Bennett et al. 1997]; a node wishing to communicate must stay awake listening for a broadcast message announcing its neighbor's reactivation. In S-MAC [Ye et al. 2002b; 2003], nodes are locally synchronized to follow a periodic listen and sleep scheme.

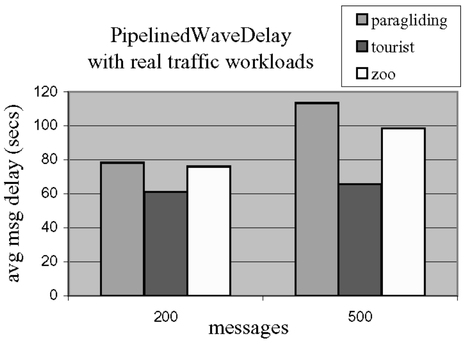

Fig. 30. Real traffic workloads: delay.

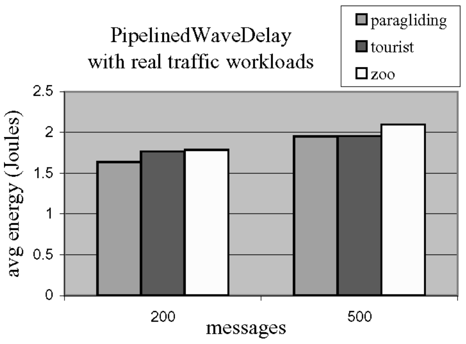

Fig. 31. Real traffic workloads: energy.

S-MAC does not explicitly avoid contention for the medium, but reduces the period of overhearing by annotating long DATA packets with their size. NAMA and TRAMA avoid all collisions at the MAC layer by announcing the schedules of nodes in the 2-hop neighborhood and electing nodes to transmit in a given time slot. Our waves avoid schedule propagation overhead, at the expense of having fixed slots for every edge activation.

In a recent independent work, researchers have studied the problem of sleep scheduling in sensor networks to minimize end-to-end communication delay, and have obtained a number of complexity results and heuristics for trees, grids, and general graphs [Lu et al. 2005]. The particular problem formulation, however, is different than ours. Their focus is on periodic schedules in which each node is activated once every $k$ slots, for a given parameter $k$, while we allow more general periodic schedules. Furthermore, in our study, we also consider the interaction of activation scheduling and routing in the design of energy- and delay-optimal communication schemes.

**Routing algorithms:** Several routing protocols for ad hoc networks have been proposed in the literature [Perkins and Bhagwat 1994; Johnson and Maltz 1996; Broch et al. 1998; Perkins 1999; Park and Corson 1999]. There has also been a plethora of work on energy-aware routing [Chang and Tassiulas 2000; Singh et al. 1998; Yu et al. 2001] but without considering the interplay of routing and

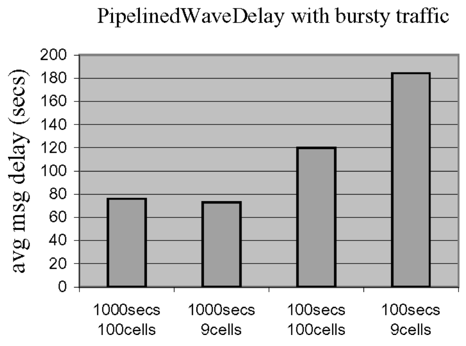

Fig. 32. Bursty traffic workloads: delay.

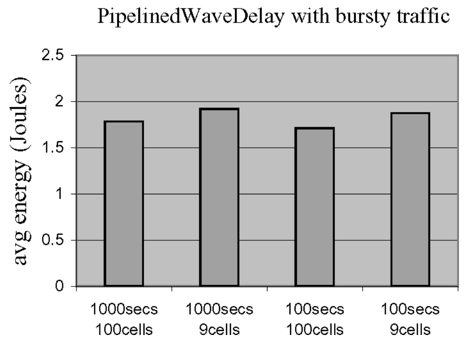

Fig. 33. Bursty traffic workloads: energy.

scheduling. The TinyDB Project at Berkeley investigates tree-based routing and scheduling techniques for sensor networks [Madden et al. 2002; Hellerstein et al. 2003]. An energy-efficient aggregation tree using data-centric reinforcement strategies is proposed in Intanagonwiwat et al. [2000]. A two-tier approach for data dissemination to multiple mobile sinks is discussed in Ye et al. [2002a].

**Radio Model Studies:** Ganesan et al. [2002] present an empirical study involving 150 low-power sensor nodes, in which they discuss the directionality and asymmetry of radio propagation. Their finding that at short distances from the transmitter, a negligible precentage of links are asymmetric is consistent with our assumption that most asymmetric links are found at distances greater than half the communication range (Figures 14 and 15). Ganesan et al. [2002] also show the positive influence of transmit power on the percentage of symmetric links at a given distance. For our purposes, we could calibrate the transmit power so that the greatest percentage of symmetric links are at a cell's distance. Kotz et al. [2003] illustrate six false axioms that most experimental studies rely on, namely 1) the world is flat, 2) a radio's transmission area is circular, 3) all radios have equal range, 4) if I can hear you, you can hear me, 5) if I can hear you at all, I can hear you perfectly, and 6) signal strenth is a simple function of distance. The extent to which our algorithms rely on these axioms is discussed in Section 3.2. Zhao and Govindan [2003] report that up to a certain distance from the sender, packet reception rates are uniformly high for Mica motes. Beyond this there is a gray area in which reception rates can vary dramatically between different nodes. They also report that the packet reception rate variance is low for all receivers up to some distance, which means that most links within a cell are consistently reliable. Cerpa et al. [2003] compared connectivity conditions between the Mica 1 and Mica 2 platforms. The great majority of Mica 2 motes transmitting at medium-high power in an outdoor habitat exhibited very high delivery probabilities at all distances within range. This would be a very suitable scenario for running our algorithms. However, Mica 1 motes transmitting at medium power in an indoor office exhibit great variation in reception rate for almost all distances tested, making our algorithms unsuitable in this case. Aguayo et al. [2004] perform a similar study, but in a 802.11b mesh network. They report that most nodes that can communicate have intermediate loss rates, and that the correlation of distance and delivery probability is weak. Our algorithms are not suitable for environments where the majority of links within a cell or across adjacent cells are either asymmetric or unreliable. Aguayo et al. [2004] show that there is considerable difference from link to link in the burstyness of the delivery probability, and report that only a significant *minority* of links varies substantially in loss rate from one second to the next. The consistency of link quality over time is important, as it suggests that stationary networks with few moving obstacles will rarely need to repair established routes. Zhou et al. [2006] also discuss the irregularities of radio propagation (anisotropy, continuous variation, and different sending powers) and propose the RIM model to approximate them. They study the impact of radio irregularities on MAC and routing protocols (for a similar discussion specific to our algorithms, see Section 3.2).

## 8. CONCLUSIONS AND FUTURE WORK

In this article, we have presented a class of algorithms that allow us to trade energy versus delay for data dissemination in sensor networks. Our approach is based on carefully *scheduling* the sensor nodes so that each node can stay idle most of the time, turning on its radio only at scheduled intervals during which it can receive or send messages. Although it is easy to combine an arbitrary routing protocol with the proposed wave schedules, the former may not fully leverage the potential of wave schedules for collision avoidance. Our routing protocol is designed to minimize interference, when combined with a wave schedule, while remaining robust to localization errors and radio irregularities. Our experiments show that the proposed scheduling and routing algorithms result in significant energy savings at the expense of increased message latency.

In the future, we plan to study irregular wave schedules, in which we relax the current assumption that every directed edge in the network is scheduled regularly once per period, and thus has the same capacity. In practice, incoming edges to view nodes are expected to be more heavily loaded than edges at the border of the network. We believe that better network utilization can be achieved by considering a more general class of wave schedules in which different edges are activated with different rates. For instance, the network can be divided into highways (frequently activated edges) and driveways (low capacity edges). It would be interesting to study the tradeoff between energy and delay in such an irregular model.

## ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers and the editor for their valuable comments.

## REFERENCES

Aguayo, D., Bicket, J., Biswas, S., Judd, G., and Morris, R. 2004. Link-level measurements from an 802.11b mesh network. In *Proceedings of the Annual Conference of the Special Interest Group on Data Communication (SIGCOMM)*. 121–132.

Bahl, P. and Padmanabhan, V. N. 2000. RADAR: An in-building RF-based user location and tracking system. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*. 775–784.

Bennett, F., Clarke, D., Evans, J., Hopper, A., Jones, A., and Leask, D. 1997. Piconet: Embedded Mobile Networking. *IEEE Pers. Comm. 4,* 5 (Oct.), 8–15.

Breslau, L., Estrin, D., Fall, K., Floyd, S., Heidemann, J., Helmy, A., Huang, P., McCanne, S., Varadhan, K., Xu, Y., and Yu, H. 2000. Advances in network simulation. *IEEE Compu. 33,* 5 (May), 59–67.

Broch, J., Maltz, D. A., Johnson, D. B., Hu, Y.-C., and Jetcheva, J. 1998. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*. 85–97.

Bulusu, N., Heidemann, J., and Estrin, D. 2000. Gps-less low cost outdoor localization for very small devices. *IEEE Pers. Comm. Mag. 7,* 5 (October), 28–34.

Cerpa, A., Busek, N., and Estrin, D. 2003. Scale: a tool for simple connectivity assessment in lossy environments. Tech. rep., UCLA, CENS-21.

Chang, J.-H. and Tassiulas, L. 2000. Energy conserving routing in wireless ad hoc networks. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*. IEEE, Los Alamitos, 22–31.

Chen, B., Jamieson, K., Balakrishnan, H., and Morris, R. 2002. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *ACM Wireless Networks 8,* 5 (September).

Clark, B., Colbourn, C., and Johnson, D. 1990. Unit disk graphs. *Discrete Mathematics 86*, 165–177.

Crossbow Mica2. 2005. Datasheet of the mica2 (mpr400cb) sensor node platform. Crossbow Technology. Inc.

Elson, J., Girod, L., and Estrin, D. 2002. Fine-grained network time synchronization using reference broadcasts. *ACM SIGOPS Operating Systems Review, SI: Physical Interface 36*, 147–163.

Ganeriwal, S., Kumar, R., and Srivastava, M. 2003. Timing-sync protocol for sensor networks. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (SENSYS)*. 138–149.

Ganesan, D., Krishnamachari, B., Woo, A., Culler, D., Estrin, D., and Wicker, S. 2002. Complex behavior at scale: An experimental study of low-power wireless sensor networks. Tech. Rep., UCLA, CSD-TR 02-0013. February.

Garey, M. and Johnson, D. 1977. The rectilinear Steiner tree problem is NP-complete. *SIAM J. Appl. Math. 32*, 826–834.

Garey, M. R. and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York.

Ghose, A., Grossklags, J., and Chuang, J. 2003. Resilient data-centric storage in wireless ad hoc sensor networks. In *Proceedings of the 4th International Conference on Mobile Data Management (MDM)*. 45–62.

Hellerstein, J., Hong, W., Madden, S., and Stanek, K. 2003. Beyond average: Towards sophisticated sensing with queries. In *2nd International Workshop on Information Processing in Sensor Networks (IPSN)*.

Intanagonwiwat, C., Govindan, R., and Estrin, D. 2000. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM)*. 56–67.

Johnson, D. B. and Maltz, D. A. 1996. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, Imielinski and Korth, Eds. The Kluwer International Series in Engineering and Computer Science, vol. 353. Kluwer Academic Publishers.

Kasten, O. 2001. Energy consumption. Tech. rep., Eldgenossische Technische Hochschule (ETH) Zurich. `http://www.inf.ethz.ch/˜kasten/research/bathtub/energy_consumption.html`.

Kotz, D., Newport, C., and Elliott, C. 2003. The mistaken axioms of wireless network research. Tech. Rep., Dartmouth College Computer Science, TR2003-467. July.

Langendoen, K. and Reijers, N. 2003. Distributed localization in wireless sensor networks: a quantitative comparison. *Compu. Netw. 43,* 4, 499–518.

Lu, G., Sadagopan, N., Krishnamachari, B., and Goel, A. 2005. Delay efficient sleep scheduling in wireless sensor networks. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*.

Madden, S. R., Franklin, M. J., Hellerstein, J. M., and Hong, W. 2002. TAG: A tiny aggregation service for ad hoc sensor networks. In *Proceedings of the Symposium on Operating System Design and Implementation (OSDI)*.

Park, V. and Corson, S. 1999. Temporally-ordered routing algorithm (tora) version 1 functional specification. Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-manet-tora-spec-02.txt.

Perkins, C. and Bhagwat, P. 1994. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proceedings of the Annual Conference of the Special Interest Group on Data Communication (SIGCOMM)*. 234–244.

Perkins, C. E. 1999. Ad hoc on demand distance vector (aodv) routing. Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-04.txt.

Ratnasamy, S., Karp, B., Yin, L., Yu, F., Estrin, D., Govindan, R., and Shenker, S. 2002. Ght: A geographic hash table for data-centric storage. In *First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*.

Sen, A. and Huson, M. 1996. A new model for scheduling packet radio networks. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*. 1116–1124.

Singh, S., Woo, M., and Raghavendra, C. S. 1998. Power-aware routing in mobile ad hoc networks. In *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*. 181–190.

Society, I. C. 1999. Wireless LAN medium access control (mac) and physical layer specification. IEEE Std 802.11.

Stemm, M. and Katz, R. 1997. Measuring and reducing energy consumption of network interfaces in hand-held devices. *IEICE Trans. Comm. E80-B*, 1125–1131.

Trigoni, N., Yao, Y., Demers, A., Gehrke, J., and Rajaraman, R. 2004. Wavescheduling: Energy-efficient data dissemination for sensor networks. In *International Workshop on Data Management for Sensor Networks (DMSN)*.

Werner-Allen, G., Tewari, G., Patel, A., Welsh, M., and Nagpal, R. 2005. Firefly-inspired sensor network synchronicity with realistic radio effects. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (SENSYS)*. 142–153.

Xu, Y., Heidemann, J., and Estrin, D. 2001. Geography-informed energy conservation for ad hoc routing. In *Proceedings of the International Conference on Mobile Computing and Networking (MOBICOM)*. 70–84.

Ye, F., Luo, H., Cheng, J., Lu, S., and Zhang, L. 2002. A two-tier data dissemination model for large-scale wireless sensor networks. In *Proceedings of the International Conference on Mobile Computing and Networking (MOBICOM)*.

Ye, W., Heidemann, J., and Estrin, D. 2002. An energy-efficient MAC protocol for wireless sensor networks. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*. 1567–1576.

Ye, W., Heidemann, J., and Estrin, D. 2003. Medium access control with coordinated, adaptive sleeping for wireless sensor networks. Tech. Rep. ISI-TR-567, USC/Information Sciences Institute. January.

Yu, Y., Govindan, R., and Estrin, D. 2001. Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks. Tech. Rep. UCLA/CSD-TR-01-0023, University of Southern California. May.

Zhao, J. and Govindan, R. 2003. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (SENSYS)*. 1–13.

Zhou, G., He, T., Krishnamurthy, S., and Stankovic, J. 2006. Models and solutions for radio irregularity in wireless sensor networks. *ACM Trans. Sen. Netw. 2,* 2, 221–262.

Received May 2006; revised April 2006, October 2006; accepted October 2006