Dozer: Ultra-Low Power Data Gathering in Sensor Networks

Nicolas Burri
Computer Engineering and Networks Laboratory
ETH Zurich
8092 Zurich, Switzerland
nburri@tik.ee.ethz.ch

Pascal von Rickenbach
Computer Engineering and Networks Laboratory
ETH Zurich
8092 Zurich, Switzerland
pascalv@tik.ee.ethz.ch

Roger Wattenhofer
Computer Engineering and Networks Laboratory
ETH Zurich
8092 Zurich, Switzerland
wattenhofer@tik.ee.ethz.ch

ABSTRACT

Environmental monitoring is one of the driving applications in the domain of sensor networks. The lifetime of such systems is envisioned to exceed several years. To achieve this longevity in unattended operation it is crucial to minimize energy consumption of the battery-powered sensor nodes. This paper proposes *Dozer*, a data gathering protocol meeting the requirements of periodic data collection and ultra-low power consumption. The protocol comprises MAC-layer, topology control, and routing all coordinated to reduce energy wastage of the communication subsystem. Using a tree-based network structure, packets are reliably routed towards the data sink. Parents thereby schedule precise rendezvous times for all communication with their children. In a deployed network consisting of 40 TinyOS-enabled sensor nodes, *Dozer* achieves radio duty cycles in the magnitude of 0.2%.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; C.2.2 [**Computer-Communication Networks**]: Network Protocols

General Terms

Algortithms, Measurement

Keywords

Sensor network, data gathering, energy efficiency

1. INTRODUCTION

Observation and interpretation of natural phenomena has always been of fundamental importance to numerous research areas. Sensor networks represent a tool which provides the possibility to sample and gather data at scales and resolutions which were difficult to obtain before. By spreading large numbers of cheap untethered sensor nodes in an area of interest scientists are enabled to monitor dense temporal and spatial data over an extended period of time. With this data the analysis of complex interactions becomes possible; this task is also known as environmental monitoring. Thus, sensor networks have the potential to support the advancement of various fields of research.

Wireless sensing devices exhibit a large variety of favorable attributes. They facilitate the deployment and are far less intrusive than tethered solutions. Furthermore, they permit temporary measurements or surveillance of secluded areas. In addition sensor networks should not need any human interaction while fulfilling their intended tasks. Due to the limited capacity of common power supplies for sensor networks, such as batteries or solar cells, energy efficiency is a fundamental requisite for prolonged network lifetime. All sensor nodes are equipped with a short-ranged radio allowing them to convey their data to an information sink for further processing. This communication subsystem is one of the primary power consumers of a sensor node. The energy wastage of the radio, even in idle listening, is three orders of magnitude higher than a node's power drain in sleep mode. As a consequence, the radio should only be turned on if a data transfer is pending. This requirement is hard to fulfill since multi-hop routing techniques must be applied to transmit data from all nodes in a possibly large area to the data sink. Energy-efficient data exchange is a nontrivial task in single-hop networks but becomes even more challenging if routing over multiple hops is required. Sensor nodes are no longer able to schedule their transmissions strictly according to their individual demands but they also have to activate their radio in order to receive and relay messages from other nodes in the network. This raises the well-known problems of idle listening and overhearing which waste precious energy.

In this paper we consider applications in the field of environmental monitoring—also known as data gathering—for wireless sensor networks. We thereby focus on applications producing continuous data. Examples thereof include precision agriculture [1], glacier displacement measurements [12], natural habitat monitoring [11], or microclimatic observations [17]. All of these applications generate periodic data samples at low rates resulting in light traffic load and thus low bandwidth requirements. We propose Dozer, an ultra-low power network stack tailored for data gathering applications. It incorporates a MAC layer, topology control, and a routing protocol. We refrained from integrating existing low-power solutions for any of these subsystems since it is our strong belief that only a perfectly orchestrated network

IPSN'07, April 25-27, 2007, Cambridge, Massachusetts, USA.

stack is able to achieve minimum power consumption and therefore maximize network lifetime. The primary goal of Dozer is to reduce idle listening and overhearing. In theory, a TDMA-based MAC protocol constructing a global schedule to determine exact send and receive times for each node would solve the problem of overhearing and idle listening. In a real-world setting clock drifts and frequently changing external conditions render plain TDMA costly since maintaining an accurate schedule is a complex and energy consuming task. Dozer takes these actualities into account. It builds a data gathering tree on top of the underlying network topology and provides nodes with precise wakeup schedules for all communication only relying on local synchronization. Furthermore, it addresses the problem of temporary network partition and energy efficient tree adaptation in case of local link failures. Despite these additional considerations Dozer attains low radio duty cycles in both single-hop and multi-hop networks and thus achieves high energy efficiency. The protocol was implemented using TinyOS. Its performance was evaluated using an indoor deployment consisting of 40 TinyNode [4] sensor nodes. Using a sampling period of two minutes Dozer achieved an average duty cycle of less than 0.2% on all nodes. Given two off-the-shelf AA-sized lithium batteries with a capacity of 2000 mAh each and ignoring power consumption of application specific sensor equipment, as well as battery self discharge, Dozer is able to operate the network for a lifetime of approximately 5 years. As a consequence, system lifetime is determined by the self-discharge rate of modern batteries. To the best of our knowledge this also makes Dozer one of the most energy efficient multi-hop data gathering systems which have been designed and implemented to date.

The remainder of the paper is organized as follows: After discussing related work in the next section, we give an overview of Dozer in Section 3. Section 4 describes the implementation details of our system. In the subsequent section we present an experimental evaluation of Dozer. Section 6 concludes the paper.

2. RELATED WORK

Corresponding to the importance of the problem, there have been a plethora of research efforts addressing data gathering in the last few years. The energy efficiency of most existing work [8, 11, 17, 15, 2] stems from the application of generic energy-efficient MAC protocols [13, 18, 20]. These protocols turn off the wireless transceiver whenever possible to save power. Two types of protocols are thereby distinguished: TDMA and contention-based protocols. Protocols falling in the latter category incorporate duty cycling to achieve low power operation. [20] and [10] coordinate the nodes' sleep schedules such that neighboring nodes are awake at the same time. In the active phases CSMA/CA is used to control channel access. To achieve high energy efficiency the active periods must be very small compared to the time nodes are in sleep mode. Since the whole network wakes up at roughly the same time nodes suffer from high channel contention which reduces network throughput. T-MAC [18] is an improvement of S-MAC [20] handling varying traffic load with adaptive duty cycling. The protocol does however not overcome the inherent limitations of this approach. *Low-power listening* is another strategy to condition contention based MAC protocols to low-power requirements. To avoid idle listening nodes turn off the radio most of time, only periodically probing the channel for the presence of activity. Once network activity is detected the node switches on its radio to listen for the incoming packet. To ensure the receiver is listening a sender has to prefix its packet with a long preamble acting as an in-band busy-tone. A key advantage of asynchronous low-power listening protocols [13, 3] is that the sender and receiver can be completely decoupled in their own duty cycles. However, these protocols suffer from the overhearing problem, since the long preamble also wakes up nodes who are not the intended receiver of a packet. To overcome this drawback [21] proposes to synchronize the channel polling times of all neighboring nodes, thus preventing the protocol from sending long preambles. This move incurs contention during the scheduled channel probing which is resolved by using CSMA. A drawback of this protocol is that all nodes require to be tightly synchronized to meet energy efficiency which creates additional costs.

In contrast to the aforementioned protocols, TDMA-based solutions establish a schedule where each node is assigned one or possibly multiple time-slots. In each slot nodes are then able to communicate without provoking packet collisions or suffering from overhearing. Pure TDMA protocols are however hardly feasible in reality since they require global time synchronization and are susceptible to topological changes of the network. Hence, most proposed protocols use a combination of pure TDMA and the above mentioned contention-based approach.

In [14] a two phase protocol is proposed. In the first phase a node collects information about its two-hop neighborhood and participates in a distributed slot allocation procedure. In addition, a protocol for network-wide time synchronization is executed during this phase. Once the TDMA schedule is computed in the first phase the protocol switches over to the second phase where the schedule is executed. DMAC [9] proposes an adaptation of S-MAC optimized for data gathering. The protocol assumes that a routing tree towards the data sink exists. The active periods of the nodes are staggered according to their level in the tree. CSMA is used to arbitrate between children in order to prevent collisions. DMAC achieves low data delivery latency at the sink. However, there is a substantial overhead in case of network instabilities and due to the local synchronization at the nodes. FPS [6] and its descendant Twinkle [5] are closest related to the protocol described in this paper. The coarse grained scheduling of FPS represents a distributed TDMA approach where each node schedules its own children. Although this schedule ensures that parents and their children are contention free, collisions may still occur due to other nodes in the network or poor time synchronization. This contention is handled using CSMA. The protocol does not incorporate a tree construction and is thus dependent on other protocols establishing such a network topology. In contrast to our solution FPS—and thus also Twinkle—requires global time synchronization.

Another branch of research in the field of data gathering explores in-network data processing also known as data aggregation or data fusion [7, 16, 19, 22, 23]. However saving energy by reducing the amount of data actually sent to the base station is orthogonal to our work and goes beyond the scope of this paper.

3. DOZER OVERVIEW

The Dozer system is indented to meet common demands of environmental monitoring applications. It enables reliable data transfer, has self-stabilizing properties—and is thus robust to changes in the environment—, and it is optimized for long system lifetime. Network latency and flexibility towards dynamic bandwidth demands are considered to be of less importance.

In order to forward data to the base station Dozer establishes a tree structure on top of the physical network. This guarantees that information from any node is conveyed on a loop-free path to the data sink which constitutes the root of the tree. Each node fills two independent roles in tree maintenance. On the one hand, it acts as a parent for directly connected nodes one level deeper down the tree. On the other hand, it is a child of exactly one node one level higher in the tree. Data is transfered to the sink using a TDMA protocol. However Dozer does not construct one global schedule for the whole network but splits it up at each node. Consequently, each node has two independent schedules; one in its role as a parent and one for its child role. As a parent a node decides when each of its children is allowed to upload data. Vice versa, in its role as a child it receives an update slot from its own parent. Thus, Dozer only constructs single-hop schedules and does not rely on any global synchronization. Each round of a parent's TDMA schedule is initiated by the transmission of a beacon message. Simplified, beacons are the heart beat of the Dozer system. In its child role, a node synchronizes on the received parent beacon. However, it does not adjust its internal clock but calculates the outset of its upload slot in relation to the last beacon reception time.

Dozer does not make use of a traditional MAC protocol. In fact, the system does not try to prevent nodes from sending at the same time; collisions are explicitly accepted (c.f. in Section 4.2). However, using randomization Dozer ensures that two schedules drift apart quickly in case of a collision. This scheme is advantageous as a message receiver always exactly knows when the corresponding sender is going to start its transmission. This greatly prolongs network lifetime since nodes are able to maximize their time in energy-efficient sleep mode. Facing collisions data transmissions in Dozer are always explicitly acknowledged.

As network conditions change over time so does the network topology. Consequently, the data gathering tree cannot be stable in the long run. To reduce increased message delay in case of link failures, each node maintains a list of additional potential parents. Choosing a candidate from this list a new connection can be established with little overhead.

4. DOZER IMPLEMENTATION

The high-level overview in Section 3 outlines that Dozer handles several interwoven tasks in parallel. More precisely, the system can be subdivided into four logical components. Figure 1 depicts the individual components and shows how they interact with each other. In the following the function of each component is discussed in detail.

4.1 Tree Maintenance

The *Tree Maintenance* module coordinates a node's integration in the data gathering tree of Dozer and guarantees

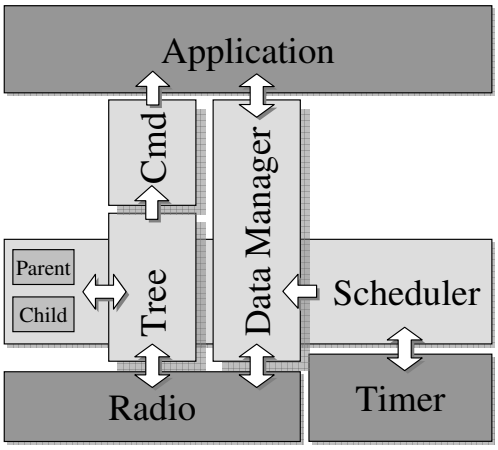

Figure 1: Architecture of the Dozer system represented by the light gray boxes. Arrows indicate the command flow between the different modules.

constant connectivity. Furthermore, in case of a network failure it sets the node in an energy efficient suspend mode until a reintegration in the tree becomes possible.

4.1.1 Connection Setup

It is essential for every node to be part of Dozer's data gathering tree. Nodes without connectivity are unable to provide data to the base station and are thus of no use. Upon wakeup, in the bootstrap phase, a node tries to join the tree as quickly as possible. Since it does not yet have any conception of its neighborhood, it starts listening for beacon messages of nearby nodes. Beacon messages are periodically sent by already connected nodes at the beginning of their TDMA schedule to enable the integration of disconnected nodes. After scanning for the full length of a TDMA round each received beacon message is analyzed and the corresponding node is ranked according to a rating function. The function's current implementation considers a node's distance to the sink as well as its load—the number of direct children—in this computation. Both of these values are part of the beacon message and are thus readily available. To minimize tree depth, distance has a higher weight than load in the computation. The node now tries to connect to the highest rated neighbor and the gathered information about all other overheard potential parents is stored.

The actual connection setup is initiated after the transmission of the next beacon of the selected neighbor (see Figure 2). After sending its beacon the potential parent stays in receive mode for a short amount of time. Within this contention window it accepts incoming connection requests. The child uses a simple back-off mechanism to determine when to send its connection request message. This contention phase is needed since multiple nodes may want to establish a connection with this parent at the same time. On receiving a connection request message the parent assigns the new child a slot in its TDMA schedule and returns this informations by means of a handshake message. Currently a node only accepts one new child per beacon interval. This restriction serves as a simple form of load balancing. A node failing to connect to a specific neighbor may first try

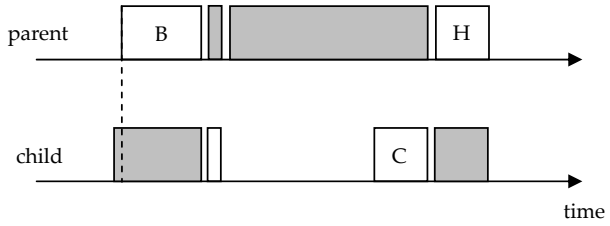

Figure 2: Connection setup – The parent node sends a beacon (B). Upon beacon reception the child sends a busy tone to activate the contention window. The child then transmits its connection request (C). A handshake (H) serves as an acknowledgment. Shaded areas denote the times a node is actually listening.

to join the tree at another node with similar rating before retrying on the same parent.

Since listening for the whole length of the contention window after each beacon transmission is expensive, in Dozer an activation mechanism precedes the actual connection setup. As depicted in Figure 2 the child transmits an *activation frame* immediately after receiving the potential parent's beacon message. On the other hand, the parent switches to receive mode and polls the channel right after sending its beacon. Only if the received radio signal strength (RSSI) indicates channel activity the contention phase is activated. If multiple nodes want to connect to the parent in the same round their activation frames collide.This imposes no problem since the parent does not try to detect a specific pattern and the sensed RSSI still clearly indicates activity on the medium.

4.1.2 Connection Recovery

Wireless links are fragile to changes in the environment and must be expected to break at any time. Unstable weather conditions or temporary obstacles in the area of interest can have a negative impact on the network stability. Dozer incorporates a mechanism to confront this problem.

A connection to the current parent breaks if multiple consecutive data transfers fail: The parent is declared unreachable. To replace it with little overhead, the orphaned node queries its stored list of potential parent for a well suited substitute and tries to establish a new connection. In case of success this procedure costs a reasonably small amount of energy. However, if no replacement can be found in this list the node falls back to bootstrap mode (see Section 4.1.1) and has to conduct a costly scan in order to detect new potential parents. To guarantee the availability of reasonably up-to-date information about its stored potential parents, a node periodically listens for their beacons. This refresh is cheap since future beacon transmission times of a node can be predetermined accurately based on the point in time of its last overheard beacon. This calculation is performed by the Scheduler component described in Section 4.2. Additionally, to learn about the existence of new, potentially well suited parents, a random listen mechanism is applied. Unrelated to their two schedules nodes periodically overhear the channel for beacons of yet unknown nodes. To keep the incurred overhead low, these scans must only be executed infrequently.

4.1.3 Suspend Mode

If a node is not connected to a parent and also cannot hear any beacons—even when listening for a full beacon interval—it assumes the network to be down or out of reach. Constant channel surveillance in this situation would result in high power consumption; a node's lifetime would decrease to a couple of days. To circumvent such energy wastage Dozer features a special suspend mode. Along the line of low power listening [13] the node periodically samples the channel for activity and remains in sleep mode for the remainder of the time. However, unlike low power listening this mode does not ensure reception of all messages. Energy efficiency and quick reactivation in case of channel usage can be balanced depending on the demands of the application running on top of Dozer. Frequent channel polling results in higher power drain but rapid connection establishment on network availability. On the other hand, longer intervals between scans lead to improved energy efficiency but possibly delayed reintegration of suspended nodes.

4.2 Scheduler

The energy efficiency of the Dozer system mostly stems from the *Scheduler* module. By providing the Tree Maintenance and Data Manager modules with precise timings it allows efficient radio usage.

Communication between a parent and its children is coordinated by a TDMA protocol. That is, all transmissions happen at exactly predetermined moments in time. For the exchange of a message neither sender nor receiver have to spend energy beyond what is required to transmit or receive the actual data. In particular nodes do not have to waste energy on overhearing the channel for pending transmissions. A global TDMA scheme however is expensive since it demands the existence of a network-wide time synchronization mechanism. To circumvent this burden Dozer only aligns one hop neighbors in the data gathering tree. As all nodes are simultaneously parent and child they all have to maintain two schedules; one provided by their parent and one self-determined as a reference for their children. In this setting it is complex to synchronize the internal clocks of a parent and its children. Only by means of global time synchronization it would be possible for each node to service both schedules with only one clock.

While in theory wakeup times can be calculated perfectly at both parent and children, clock drift has to be considered in real-world applications. The current generation of sensor nodes is usually equipped with an electronic oscillator exhibiting a skew of 30-50 parts per million (ppm) at room temperature. Thermal differences between sender and receiver lead to significant, additional skew. In Dozer, the receiver of a transmission is responsible for clock drift compensation and worst-case guard times are used to guarantee a prior wake up of the receiver before the sender starts its transmission.

The self-determined TDMA schedule of a node, in the following also denoted as parent schedule, is of fixed length and divided into equal time slots. Upon connection of a new child the Tree Maintenance module requests a free slot from

the Scheduler. This slot is henceforth marked as occupied and reserved for the new child. The assignment outlasts the end of the schedule and is only released if the corresponding child disconnects. That is, each child owns the same time slot in every iteration of the schedule. As a consequence, the total number of slots of the TDMA schedule confines the maximum number of children a node is able to manage.

After connection establishment between parent and child, the personal slot number of the child in its parent's schedule is known at both nodes. They can thus compute the start of this slot relative to the beginning of the schedule. For each slot of its schedule the parent checks if it is occupied and listens for incoming data if necessary. Analogously, at the child the Scheduler triggers the Data Manager component at the start of its upload slot to permit a timely transfer of potentially queued data messages.

As mentioned above the protocol does not provide for any direct clock synchronization. Instead, at the outset of a new round of the schedule the Tree Maintenance module is triggered to send a beacon message. This beacon is received by all children and timestamped according to their local clocks. Since both parent and children share the knowledge about the time of the beacon transmission this moment in time serves as an anchor point for implicit clock synchronization. No adjustments of system clocks are required but only this timestamp needs to be stored for further timing calculations. For the remainder of this round of the TDMA schedule all events are computed in relation to this timestamp. The transmission time of the next beacon is also determined according to this value. As a positive side effect, clock drift accumulation over multiple rounds of the schedule is prevented. Furthermore, the complexity of handling both independent schedules diminishes since only two values, used as offsets for the internal clock need to be stored.

Without a global schedule, collisions between the transmissions of neighboring nodes that are not part of the same schedule can no longer be excluded. Other systems facing the same problem (e.g. [5]) apply secondary MAC protocols such as CSMA/CA to resolve it. However, since bandwidth demands in the considered scenario are low, collisions happen infrequently. Dozer thus refrains from handling them actively. In the long run the costs for retransmissions are cheaper than the costs it would take to prevent them. But regarding collisions there exists an additional problem which needs to be tackled. Collisions may indicate the undesired alignment of two independent schedules. If this is the case, without intervention, collisions would recur in subsequent rounds of the schedule. To counter this threat, Dozer extends the length of a TDMA round by a randomly chosen time span—also referred to as jitter. The parent draws a new random number for each round of the schedule which is then added to the round's length. There is a linear relation between the maximum transmission time per slot and a reasonable upper bound for this random offset. Dozer uses a bound of seven times the time needed to flush the local message buffer (c.f. in Section 4.3). With this value, in case of a collision between two unsynchronized transmissions, the chance for a second consecutive collision is less than 50% in expectation. For any realistic scenario this implies that a maximum jitter of less than one second suffices.

This random prolongation of the TDMA rounds introduces the problem of how to predict the exact time of the next parent beacon. Thus, the seed value of the random number generator used for calculating the next random offset is included in each beacon. With this value each child is able to execute the same computation as the parent and to predict when the next beacon message is due. At the parent, the current random number is used as seed value to generate the next random number. Consequently, even if a child misses one or more consecutive beacon messages of its parent it is still able to determine the next beacon arrival time. Based on the information of the last beacon it has received, it recursively draws random numbers until it has compensated for the number of missed beacons.

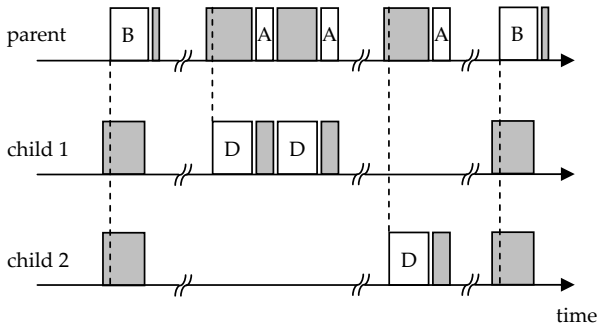

Figure 3: Message reception of a parent with two children. Upload slots are determined by parent beacon (B). All data messages (D) are explicitly acknowledged (A).

4.3 Data Administration

At the end of the day, Dozer's main task is to transport sensor readings from all nodes to the data sink. While a node's data upload times are strictly defined by the Scheduler module, data injection by the application is always possible. Hence, the *Data Manager* module features a message queue buffering injected data pending for transmission. Since data upload to the parent and data reception from the children is unsynchronized, incoming messages from the children are also appended to this queue.

As soon as the Scheduler module signals the beginning of the parent upload slot the Data Manager tries to transmit all queued messages. Each message is explicitly acknowledged and only removed from the queue of the sender if the receiver confirms its correct reception (see Figure 3). With the acknowledgment the parent not only takes over responsibility for the packet but also notifies the child about how many more messages it is willing to accept. As a consequence, at most one unnecessary message transmission is possible if the parent is unable or unwilling to handle more messages. The link acknowledgments guarantee that no messages are lost on their path towards the sink despite possible collisions on the wireless links. If a message transfer fails to be acknowledged the child immediately stops its data upload for this round of the schedule since a temporal interruption on the medium may be encountered. In case of consecutive transmission failures over multiple upload slots the Data Manager instructs the Tree Maintenance module to switch to another parent. Due to the limited amount of memory available on

current sensor node platforms the queue size is limited. Different buffering strategies may be employed depending on the application requirements. Dozer's default strategy only allows buffering of one data message from each distinct node; if more than one message from the same origin meet on a node the newer one is buffered and forwarded wheras the older one is discarded.

4.4 Command Management

While data flow in Dozer is strictly unidirectional towards the sink it is often desirable to be able to send information to one or several nodes in the network. Dozer establishes such a lightweight backward channel by making use of the beacon messages. Commands injected at the data sink are included in the sink's next beacon message. Every node receiving a beacon containing a command temporarily stores the command and includes it in its next beacon. By repeating this procedure at each level of the tree the command is disseminated through the whole network. Besides addressing a command to all nodes in the network the injection of commands for individual nodes is also supported. Nodes not directly addressed by a command still relay it to enable propagation to nodes deeper down the tree.

Upon reception of a beacon message from the parent the Tree Management component hands the command to the *Command Manager* module for further processing. The module checks if this node belongs to the set of intended recipients of the command. If this is the case the command is dispatched to the application running on top of the Dozer system. Thus, applications are able to define their own custom commands and corresponding command handlers.

5. EVALUATION

In this section we evaluate Dozer's performance under different conditions in real-world testbeds. First, a set of preliminary measurements on a small-scale network are conducted to estimate the scalability of the system. In a second step we present results of a deployed indoor network consisting of 40 sensor nodes.

5.1 Hardware and Operation System

For all experiments we used the TinyNode 584 sensor platform [4] produced by Shockfish SA. It features a MSP430 mirocontroller with 10 kB of RAM and 48 kB of program memory. Furthermore 512 kB of external flash are available. However, due to the high energy costs for access to the flash Dozer does not make use of it. The platform includes a Semtech XE1205 radio transceiver. This radio is known for its good transmission ranges and high data rates

	Current Draw	Power Consume
uC sleep with timer on , radio off	6.0 uA	0.015 mW
uC active, radio idle listening	12.17 mA	30.43 mW
uC active, radio RX	12.63 mA	31.58 mW
uC active, radio TX	16.10 mA	40.25 mW

Table 1: Measured current consumptions of the TinyNode platform in different states at 2.5 volt.

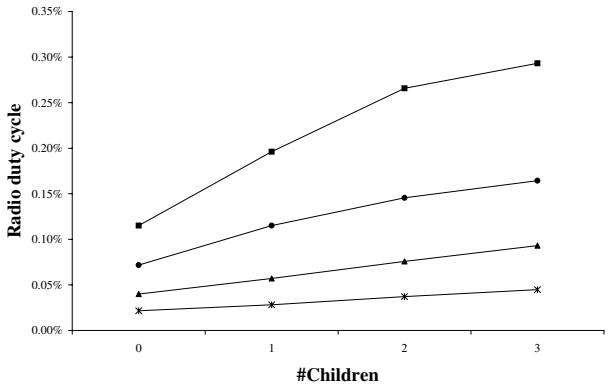

Figure 4: Radio duty cycle of a node depending on its number of children. Measurements were performed with beacon intervals of 15 s (square), 30 s (circle), 1 min (triangle), and 2 min (star), respectively.

of up to 153 kbit/s. For our measurements the nodes were operated at 868 MHz using 0 dBm transmission power and a bandwidth of 75 kbit/s[1]. As a power source two customary 1.2 volt rechargeable batteries were installed with a capacity of 1900 mAh each. The measured current draws for sleep mode, idle listening, receiving, and sending under these conditions are shown in Table 1. As can be extracted from the table, on the TinyNode platform idle listening is nearly as expensive as the actual reception of a message. Thus it benefits greatly from Dozer's scarce use of unscheduled random channel overhearing. Furthermore, the cost for transmission and reception of a message are in the same order of magnitude.

Dozer is implemented on top of the TinyOS-1.x operating system. No changes were made to the operating system excepts the replacement of a timer module whose genuine version contains a bug. Under certain conditions timer events are triggered too late. In normal operation common TinyOS-applications do not encounter this undesired behavior frequently. However, due to the heavy load on the timer module, this malfunction regularly occurs in the Dozer system with disastrous consequences. A once deferred schedule becomes useless since all relative timings are out of sync. Consequently, a node affected by this problem inevitably looses connectivity and falls back to bootstrap mode. Thus the replacement of the timer module was mission critical.

The memory footprint of Dozer is 20 kB in program memory and 1.7 kB RAM. The message queue of size 20 in the Data Manager module used as a temporary buffer for messages which need to be relayed thereby contributes 39% of the RAM usage.

5.2 Small Scale Experiments

Measuring the energy drain of a node is a non-trivial task. On the one hand, the measuring interval is too long for high-resolution measurements with an oscilloscope. On the other

[1]As described in [4], at the same transmission power, the XE1205 radio attains higher communication ranges than other state-of-the-art platforms. Hence, we are able to transmit at lower power while still achieving good ranges.

hand, a voltmeter is too inaccurate to capture short changes in current draw. Hence, we decided to measure energy consumption indirectly. For this purpose, all nodes log their radio duty cycles. This is achieved by summing up the differences between ratio startup and shutdown times. Since spotting the exact switching times from send to receive mode and vice versa is difficult, only the total uptime is recorded ignoring the specific state of the radio. This information is propagated to the base station using Dozer's own data gathering mechanism. The collected information can be converted to power consumption values using Table 1. As nodes only provide the overall radio uptime, a worst-case approximation is made. That is, it is assumed that they are always in transmit mode if their radio is active. As a consequence, all results related to energy consumption throughout the remainder of this paper can be considered as an upper bound for the actual power consumption.

To investigate the relation between a node's power drain, its number of children, and the beacon interval time we conducted a series of experiments on a small network with predefined topology. In each run the node of interest was directly connected to the sink. Over time, up to three children were included in the network and forced to connect to the monitored node. This sequence was repeated with beacon intervals in the range of 15 seconds to two minutes. The data sample interval was set to four times the length of a beacon interval. The results of these experiments are depicted in Figure 4.

Originally, the goal of this experiment was to come up with lower bounds for the achievable duty cycles at different positions in the data gathering tree. However, initial test results exhibited unexpected fluctuations when run with different sensor nodes. After closer examination, it became clear that the inherent clock drift within a single beacon interval is a significant factor influencing the total duty cycle. Thus the following results do not represent precise lower bounds. Nevertheless, they provide an accurate approximation of the radio uptimes in real networks.

Figure 4 shows that the duty cycle decreases as the beacon interval grows larger. This elementary observation is based on the fact that the number of messages to be transmitted within one beacon interval is constant independent of its length. Hence, longer intervals lead to prolonged sleeping periods without significantly increasing the radio uptime. Using a similar line of argument, the variable additional costs for a newly connected child at different beacon intervals can be understood. The reduction of the incurred overhead for the fourth child in the 15 second beacon interval experiment illustrates another phenomenon worth mentioning.

Beacon interval	30 s
Max. jitter	650 ms
Data sampling interval	120 s
Potential parents update interval	15 min
Overhearing	1 s/4 h
Compensated clock drift	100 ppm
Max. stored potential parents	5
Message queue size	20

Table 2: Configuration of the Dozer system for the office floor testbed.

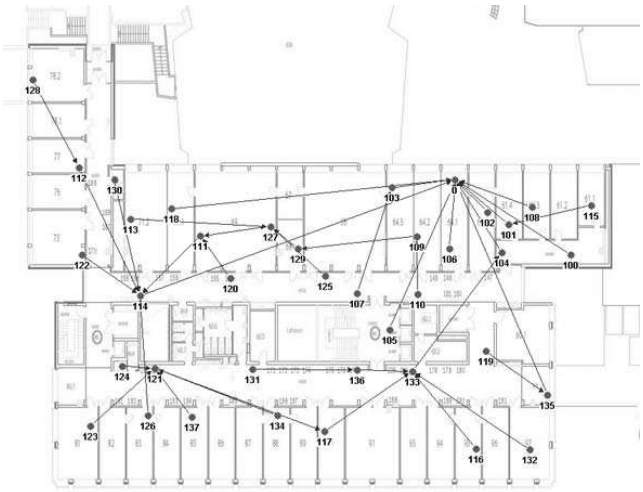

Figure 5: Indoor deployment of 40 sensor nodes including a snapshot of Dozer's data gathering tree. Node 0 (upper-right corner) acts as data sink.

Costs for additional children do not necessarily have to grow linearly. Simplified, a parent's costs for a child are twofold. On the one hand, it has to receive the child's data messages. These costs cannot be prevented. On the other hand, the parent has to forward the received messages. Thus, in its next upload slot it has to power up the radio and send the pending messages. Since the radio start-up and shutdown consumes a similar amount of time ($\sim$ 2 ms), and thus energy, as the transmission of an actual data message ($\sim$ 5 ms) its overhead is not negligible. Consequently, if the parent is able to upload data from two or more children in one upload slot it saves the additional overhead of turning on the radio for each of these children individually—costs per child decrease.

5.3 Office Floor Experiment

To put Dozer's fitness for real-world deployments to the test, a generic indoor network was run for several weeks. The topology was no longer predefined for this setting but automatically constructed by the Dozer system.

5.3.1 Setting and Protocol Parameters

The considered testbed consisted of 40 TinyNode sensor nodes which were deployed on one floor of our office building (see Figure 5). The dimensions of the building are approximately 70 x 37 meters resulting in an testbed area of more than 2500 square meters. During the whole operation of the network, the floor was populated by more than 80 people during office hours. Thus the nodes where exposed to frequently changing environmental conditions. Furthermore, during the deployment phase special attention was payed to construct a network with heterogeneous density. While nodes were concentrated in the upper-right part of the building to achieve a dense region, the southern part was only sparsely populated. This allowed the performance evaluation of Dozer in networks featuring different characteristics. In addition to 38 sensing nodes, a base station (Node 0) was placed in the upper-right corner of the map. This location

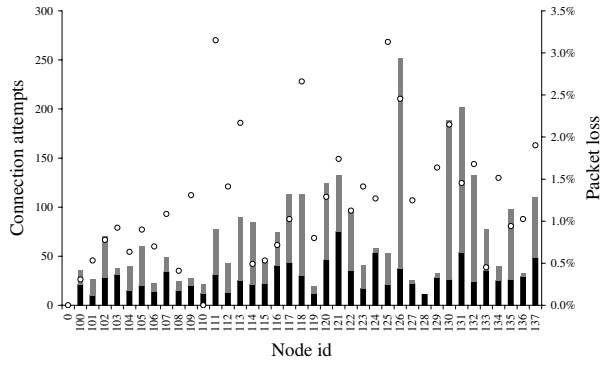

Figure 6: Number of successful (black) and failed (grey) connection attempts per node. Per node packet loss on the second y-axis.

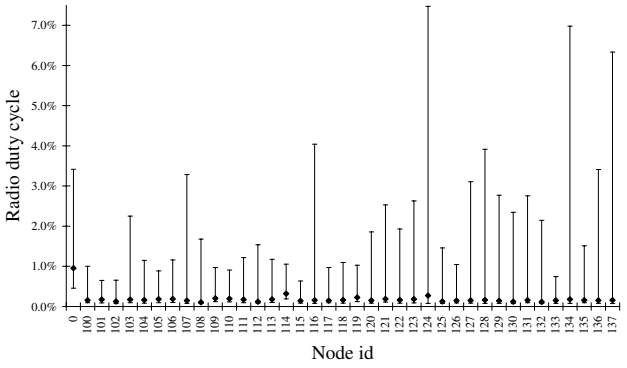

Figure 7: Average radio duty cycle of each node including RMS errors.

was chosen to get a deep data gathering tree and to enforce multi-hop communication. One further extra node was positioned in the vicinity of the sink for debugging purposes. This node acted as a network sniffer which overheard and logged all network traffic at the base station.

In total Dozer was tested for more than one month on this network. Detailed logging information forming the basis of the evaluation in this section were gathered during one week of operation. Each node thereby sent approximately 5000 data messages to the sink. As described in Section 4 the Dozer system can be tweaked to suite the requirements of a specific application. Table 2 shows the important parameters and their assigned values for the office floor testbed. Though the anticipated clock drift in our scenario is less than 50 ppm, Dozer was configured to allow for 100 ppm. Consequently, more energy than strictly necessary was consumed. In return, with these settings, the system is also expected to operate properly in outdoor environments facing moderate temperature changes. All other values were chosen to represent a possible demand of a real application in the domain of environmental monitoring.

5.3.2 Tree Topology

Figure 5 shows a snapshot of the data gathering tree as it was witnessed during the experiment. Each node features one outgoing arrow pointing to its parent. It can be seen that the base station (Node 0) has numerous children. This has two different reasons. On the one hand, the parent rating function described in Section 4.1 promotes connections to the sink since the latter has zero tree depth. As a consequence, each node receiving a sink beacon first tries to connect to the base station before inquiring any other nodes. On the other hand, the base station was flashed with a slightly modified version of Dozer. Since the sink usually runs on external power—as it is the case in our setting—it is less compelled to economize on its energy resources. Thus, the contention window was extended and the sink was configured to accept more than one child per connection phase.

Another observed phenomenon is the fact that hardly any connections passed the central core of the building. We assume that multiple sources of interference led to this barrier. For one, the corridors are lined with solid metal lockers perturbing most radio communication. On the other hand, this zone also comprises the ventilation system, sanitary facilities, and multiple elevators producing additional interference.

We examine the stability of the data gathering tree by investigating topology changes and message loss. Topology changes are indicated by a node exchanging its parent. Both of these values are depicted in Figure 6. As hoped for, message loss was low, in average 1.2% and at maximum 3.15%. However, Node 128 is excluded from this analysis. Due to its peripheral position in the network it was only able to connect to one single other node (Node 112). In case of a temporary interruption in the connection to its parent the node went to suspend mode. In addition, the low network density in its vicinity resulted in a low probability for a quick recovery. Thus, the node suffered from message loss of approximately 30%.

The measured high message yield at the base station is evidence of the correct operation of Dozer's Tree Maintenance module. As emerges from Figure 6, a significant number of topology changes were necessary to cope with momentary, local channel irregularities.

5.3.3 Energy Consumption

As in the small testbed described in Section 5.2, the energy consumption of the deployed nodes were measured indirectly via their duty cycles. Figure 7 depicts the average radio activity of each node in the network. The upward error bar shows the root mean square (RMS) error of all measurements exceeding the average duty cycle; the downward error bar is defined accordingly. The overall average duty cycle of all sensing nodes is 1.67‰ with a standard deviation of 0.0004. Applying the values from Table 1 results in a mean energy consumption of 0.082 mW.

Looking at individual nodes, the sink had by far the highest radio uptime of almost 1%. This is not surprising since it had to process the data of the whole network. Additionally, the extended contention window directly affects its duty cycle and explains the considerable difference in comparison to the sensing nodes. For further investigation of the energy consumption, we take an in-depth look at the two sensing nodes with highest duty cycles—Node 114 and Node 124. Node 124 exhibits a radio uptime of 2.8‰. Figure 8 shows a snapshot of 2000 consecutive data messages of this node. As can be seen for most of the time the node ran at a duty

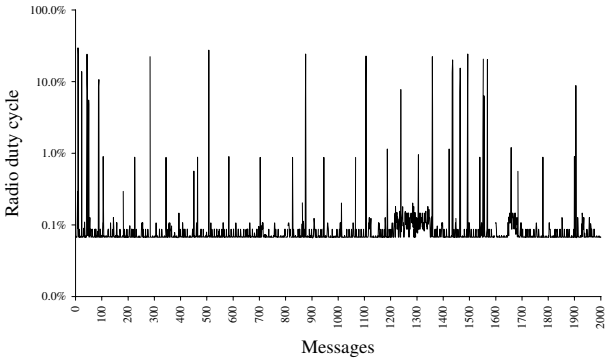

Figure 8: Radio duty cycle of Node 124 over a period of three days.

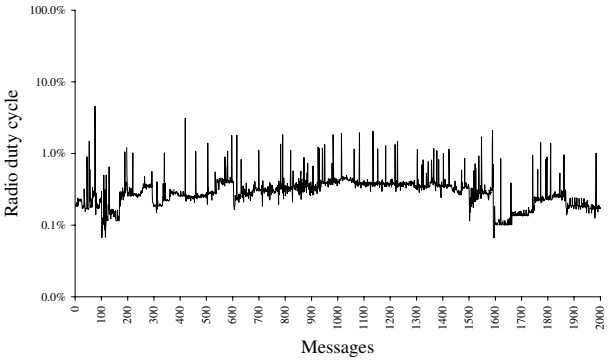

Figure 9: Radio duty cycle of Node 114 over a period of three days.

cycle of 0.7‰. Comparing this value to the results from Section 5.2 leads to the conclusion that the node is a leaf in the data gathering tree. However, three different energy intensive effects can be observed. First, the most dominant peaks exceeding 20% are scans for a full beacon interval. This means that the node was forced to establish a new connection but did not find an appropriate potential parent in its cache. Second, the overhearing phase once every four hours results in a temporary duty cycle of around 1%. Finally, the potential parents updates lead to the fringes of up to 1‰. These insights and the fact that Node 124 was located in a small storage room allows the conclusion that it had but a small neighborhood. Consequently, in times of normal operation it was able to run at nearly optimal duty cycle. However, in case of connection interruptions the interference affected all its possible connections resulting in a fallback to bootstrap mode. Unlike Node 128, it only suffered from brief network disconnections. Thus, it quickly managed to reintegrate in the data gathering tree.

Node 114 features a similar average duty cycle as Node 124, namely 3.2‰. But its power consumption is caused by other reasons as the different RMS error values indicate. Figure 9 depicts the radio duty cycle of Node 114 over a period of 2000 successive data messages. Parents updates and overhearing which are always part of a node's normal operation can also be spotted in this chart. However, there is no evidence for a bootstrap phase. In fact, Node 114 acts as a relay for several children and thus cannot reach minimal duty cycles as low as Node 124.

6. CONCLUSION

Environmental monitoring applications make great demands on wireless sensor networks. Two of the main requirements are long network lifetime and a high delivery rate of sampled sensor readings to a central authority. This paper introduces Dozer, a new data gathering system designed to meet these requirements. Using small scale experiments we have analyzed Dozer's characteristics in a controlled environment. Thereby the power consumption of individual sensor nodes dependent on their position in the established data gathering tree was measured. To verify the promising results, the system was also evaluated on an indoor testbed consisting of 40 sensor nodes. Even in this real-world deployment Dozer achieved an average radio duty cycle of 1.6‰ while guaranteeing reliable data transfer to the sink.

So far, Dozer does not try to optimize the delivery latency of sampled sensor data. We are aware that for some applications latency bounds are critical. Hence, it would be interesting to analyze how Dozer can be adapted to optimize delay. Another aspect of the protocol with potential for improvement is the compensation of clock drift. We assume that the use of an accurate clock drift prediction model would result in a further reduction of the total energy consumption. Also a precise analysis and optimization of load balancing, scalability, and fairness within the system would be of interest. Furthermore, our system currently only operates at one communication frequency. However, most modern sensor node platforms support transmissions on various channels within a frequency band. Dozer could trivially be adapted to make use of these additional communication channels and thus further reduce the collision probability between unsynchronized connections. Besides these technical improvements, it is our goal to see Dozer running on a genuine large scale sensor network collecting meaningful data.

Acknowledgements

The authors would like to thank Roger Meier and Maxime Mueller for their support in TinyNode and implementation related questions. We would also like to thank Jan Beutel for many fruitful discussions concerning the wonders of hardware and electrical engineering. Finally, we thank the anonymous reviewers for their valuable input.

7. REFERENCES

[1] T. Brooke and J. Burrell. From Ethnography to Design in a Vineyard. In *DUX '03: Proceedings of the 2003 conference on Designing for user experiences*, pages 1–4, New York, NY, USA, 2003. ACM Press.

[2] R. Cardell-Oliver, K. Smettem, M. Kranz, and K. Mayer. A Reactive Soil Moisture Sensor Network: Design and Field Evaluation. *Int. Journal of Distributed Sensor Networks*, 1(2):149–162, 2005.

[3] A. El-Hoiydi and J.-D. Decotignie. WiseMAC: An Ultra Low Power MAC Protocol for Multi-hop Wireless Sensor Networks. In *Int. Workshop on Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS)*, 2004.

[4] H. Dubois-Ferrier and R. Meier and L. Fabre and P. Metrailler. TinyNode: a comprehensive platform for wireless sensor network applications. In *Int. Conference on Information Processing in Sensor Networks (IPSN)*, 2006.

[5] B. Hohlt and E. Brewer. Network Power Scheduling for TinyOS Applications. In *EEE Int. Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2006.

[6] B. Hohlt, L. Doherty, and E. Brewer. Flexible Power Scheduling for Sensor Networks. In *Int. Conference on Information Processing in Sensor Networks (IPSN)*, 2004.

[7] B. Krishnamachari, D. Estrin, and S. Wicker. The Impact of Data Aggregation in Wireless Sensor Networks. In *Int. Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2002.

[8] K. Langendoen, A. Baggio, and O. Visser. Murphy Loves Potatoes: Experiences from a Pilot Sensor Network Deployment in Precision Agriculture. In *Int. Workshop on Parallel and Distributed Real-Time Systems (WPDRTS)*, 2006.

[9] G. Lu, B. Krishnamachari, and C. Raghavendra. An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Sensor Networks. In *Int. Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (WMAN)*, 2004.

[10] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TinyDB: An Acquisitional Query Processing System for Sensor Networks. *ACM Trans. Database Systems*, 30(1):122–173, 2005.

[11] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson. Wireless Sensor Networks for Habitat Monitoring. In *ACM Int. Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[12] K. Martinez, P. Padhy, A. Elsaify, G. Zou, A. Riddoch, J. Hart, and H. Ong. Deploying a Sensor Network in an Extreme Environment. In *IEEE Int. Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing(SUTC)*, 2006.

[13] J. Polastre, J. Hill, and D. Culler. Versatile Low Power Media Access for Wireless Sensor Networks. In *Int. Conference on Embedded Networked Sensor Systems (SenSys)*, 2004.

[14] V. Rajendran, J. Garcia-Luna-Aceves, and K. Obraczka. Energy-Efficient, Application-Aware Medium Access for Sensor Networks . In *IEEE Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, 2005.

[15] T. Schmid, H. Dubois-Ferrière, and M. Vetterli. SensorScope: Experiences with a Wireless Building Monitoring Sensor Network. In *Workshop on Real-World Wireless Sensor Networks (REALWSN)*, 2005.

[16] I. Solis and K. Obraczka. The Impact of Timing in Data Aggregation for Sensor Networks. In *IEEE Int. Conference on Communications (ICC)*, 2004.

[17] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong. A macroscope in the redwoods. In *Int. Conference on Embedded Networked Sensor Systems (SenSys)*, 2005.

[18] T. van Dam and K. Langendoen. An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *Int. Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[19] P. von Rickenbach and R. Wattenhofer. Gathering Correlated Data in Sensor Networks. In *ACM Joint Workshop on Foundations of Mobile Computing (DIALM-POMC)*, October 2004.

[20] W. Ye, J. S. Heidemann, and D. Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2002.

[21] W. Ye, F. Silva, and J. S. Heidemann. Ultra-Low Duty Cycle MAC with Scheduled Channel Polling. In *Int. Conference on Embedded Networked Sensor Systems (SenSys)*, 2006.

[22] O. Younis and S. Fahmy. An Experimental Study of Routing and Data Aggregation in Sensor Networks. In *IEEE Int. Conference on Mobile Ad Hoc and Sensor Systems (MASS*, 2005.

[23] Y. Yu, B. Krishnamachari, and V. K. Prasanna. Energy-Latency Tradeoffs for Data Gathering in Wireless Sensor Networks. In *Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2004.