Does Wireless Sensor Network Scale? A Measurement Study on GreenOrbs

Yunhao Liu, *Senior Member*, *IEEE*, Yuan He, *Member*, *IEEE*, Mo Li, *Member*, *IEEE*,
Jiliang Wang, *Member*, *IEEE*, Kebin Liu, *Member*, *IEEE*, and Xiangyang Li, *Senior Member*, *IEEE*

Abstract—Sensor networks are deemed suitable for large-scale deployments in the wild for a variety of applications. In spite of the remarkable efforts the community put to build the sensor systems, an essential question still remains unclear at the system level, motivating us to explore the answer from a point of real-world deployment view. Does the wireless sensor network really scale? We present findings from a large-scale operating sensor network system, GreenOrbs, with up to 330 nodes deployed in the forest. We instrument such an operating network throughout the protocol stack and present observations across layers in the network. Based on our findings from the system measurement, we propose and make initial efforts to validate three conjectures that give potential guidelines for future designs of large-scale sensor networks. 1) A small portion of nodes bottlenecks the entire network, and most of the existing network indicators may not accurately capture them. 2) The network dynamics mainly come from the inherent concurrency of network operations instead of environment changes. 3) The environment, although the dynamics are not as significant as we assumed, has an unpredictable impact on the sensor network. We suggest that an event-based routing structure can be trained and thus better adapted to the wild environment when building a large-scale sensor network.

Index Terms—Wireless sensor networks, network measurement, critical nodes, environment dynamics

1 Introduction

Recent advances in low-power wireless technologies have enabled us to make use of wireless sensor networks, a new class of networked systems. Researchers have envisioned a wide variety of applications, such as environment monitoring [23], scientific observation [25], emergency detection [13], field surveillance [11], and structure monitoring [28], and so on. In those applications, hundreds or even thousands of sensor nodes are assumed to be deployed in the target fields. Besides many algorithmic studies that focus on designing efficient schemes or protocols to coordinate large-scale sensor networks, there are also systematic studies that make efforts in optimizing sensor networks in practice, which are usually tested on lab-scale testbeds or small scale deployments. An essential question, however, remains unclear at the system level:

Does the wireless sensor network really scale to contain hundreds or even thousands of nodes that cooperatively work without depleting the limited physical resources, just as it was expected?

There have been several large-scale sensor network deployments reported during the past years, including Vigil-Net for field surveillance [11], Motelab that provides an indoor testbed [26], SensorScope for weather monitoring in the wild [4], and Trio which enables a large-scale solar-powered sensor network [7]. Those deployments, however, are often highly tailored to specific application and not fully leveraged as platforms for consistently observing general network behaviors. In this paper, we conduct a measurement study on GreenOrbs, which is a consistently operating sensor network system deployed for forest surveillance. With up to 330 nodes deployed in the wild, GreenOrbs provides us an excellent platform for observing sensor network behaviors at scale.

Fig. 1 plots the deployment environment and a real topology of the sensor network. The sink is deployed at the upper left corner. Each sensor node is depicted according to its 2D geographical location. We plot all the wireless links through which data packets are delivered, i.e., links used in routing protocol in our network. Nevertheless, we highlight a subgraph within the network and exhibit that the concept of "topology" does vary according to the perspective we look at it [15], [18] [21], [24]. Fig. 1a exhibits a much denser topology if we take all reachable pairs of nodes into account. Fig. 1b exhibits the topology with which the network delivers data back. Fig. 1c exhibits a topology if we select all good links that have Received Signal Strength Index (RSSI) [20] beyond a threshold. Fig. 1d exhibits a topology if we select those good links with high Link Quality Indicator (LQI) [1] when data packets are transmitted. If we consider different conditions or calibrating criterion, there will be more different types of "topologies," and thus different network performance. Indeed, as exhibited in many large-scale distributed systems, there are numerous dynamic behaviors with the concurrent and interactive operations inside the system. Such dynamic behaviors can hardly be fully considered before the system

- Y. Liu, Y. He, J. Wang, and K. Liu are with the School of Software and TNLIST, Tsinghua University, Beijing, China, 100084, and the Department of Computer Science and Engineering, Hong Kong University of Science and Technology.
E-mail: {yunhao, he, jiliang, kebin}@greenorbs.com.
- M. Li is with the School of Computer Engineering, Nanyang Technological University, N4-02C-108, 50 Nanyang Avenue, Singapore 639798.
E-mail: limo@ntu.edu.sg.
- X. Li is with Illinois Institute of Technology, 10, West 31st Street, Chicago, IL 60616. E-mail: xli@cs.iit.edu.

Manuscript received 18 March 2012; revised 20 June 2012; accepted 29 June 2012; published online 11 July 2012.
Recommended for acceptance by X. Cheng.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2012-03-0296.
Digital Object Identifier no. 10.1109/TPDS.2012.216.

1045-9219/13/$31.00 © 2013 IEEE Published by the IEEE Computer Society

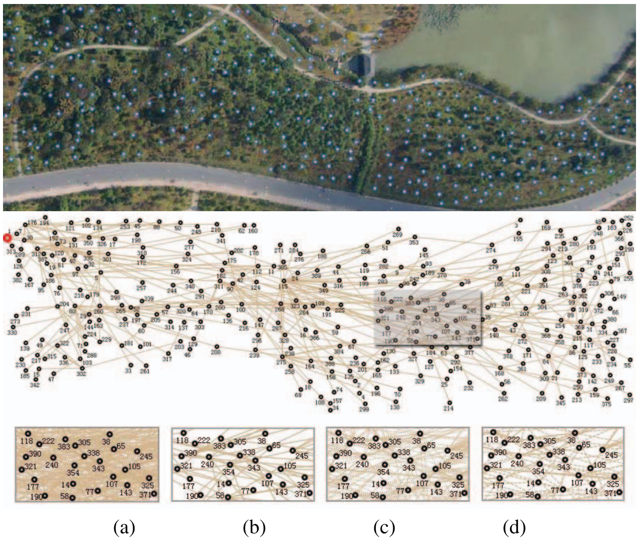


Fig. 1. An overview on the deployed sensor network. (a) The topology of reachable links. (b) The topology of data delivery link chosen by the upper layer routing protocol. (c) The topology of good quality links with RSSI thresholding. (d) The topology of good quality links with LQI thresholding when transmitting data packets.

is deployed in the field brimming of unpredictable and unexpected operating conditions.

In this paper, we conduct a measurement study on GreenOrbs, seeking to summarize the critical factors that limit the system scale out of the dynamics on the surface. We instrument such an operating network throughout the protocol stack. To obtain a comprehensive observation on the detailed interactions across different parts of the network, we collect data from the network in four separate ways:

1. Delivering network wide statistics back to the sink.
2. Deploying overhearing nodes to instrument the wireless channel.
3. Letting the sensor nodes beacon their status information when requested.
4. Locally logging the sensor node behaviors and statuses.

We vary the system settings, for example, the network scale, traffic generation, transmission power level, and test the system behaviors under a variety of conditions.

We present findings across different layers that the system works on. At the physical layer, we present measurements on radio signal strength impacted by wild environment. At the link layer, we measure packet drop/reception, link quality, transmitting rate over the entire network and how they are affected by a variety of system settings. At upper layers, we present observations on routing dynamic, traffic distribution, end-to-end packet delivery, topological features, and so on.

Our study reveals that traditional opinions on the "hot area" around sink and the instability of links may not be the only major concern for large-scale sensor network systems. The physical resources in such networks may have been underestimated and severely under utilized. There is an urgent need to improve current methods in company with those emerging critical factors when the network scales. Our experimental results also suggest us several guidelines that we should carefully consider in designing future protocols for large-scale sensor networks. In particular, the designers should take special care of the phenomena raised from the inherent contention and concurrency of numerous nodes when the sensor network scales, which might be underestimated in existing design continued from traditional WLAN or MANET.

The remainder of the paper is organized as follows: In Section 2, we describe related work in sensor network deployment and measurement experiences, as well as existing work toward making sensor networks scalable. In Section 3, we introduce the background of GreenOrbs and some details of the system implementation. In Section 4, we introduce the measurement methodology and the data trace we collected from the system. We present our major observations in Section 5. In Section 6, we give a comprehensive discussion on how the network is bottlenecked and give guidelines in mitigating such effect. We conclude this study in Section 7.

2 Related Work

In this section, we summarize the efforts of research community in building large-scale sensor networks and corresponding measurement studies.

A number of practical network deployments have been reported during the last decade. Environment and habitat monitoring has always been viewed as an important application of sensor networks [4], [10], [25]. Werner-Allen et al. [25] present a science-centric evaluation of a 19-day sensor network deployment at an active volcano, using 16 sensors to continuously sample seismic and acoustic data and initiate reliable data transmission to the base station. Tolle et al. [23] report a sensor network for monitoring the microclimate of a redwood tree. They conduct experiments on several aspects of network functions and obtain some interesting findings, for example, 15 percent of nodes die within one week by exhausting their batteries due to a problem in time synchronization. Although these findings are important, the measurements at this scale can hardly reveal some network behaviors, such as routing dynamics and topology evolution, which exist only in large-scale networks.

Researchers have designed and developed indoor medium-scale testbed such as MoteLab [26] and Kansei [8] that assist and accelerate deploying and evaluating sensor network applications. MoteLab [26] allows users to program sensors and conduct experiments remotely. Deploying sensor networks at scale is important because each order of magnitude increase in network size ushers in a new set of unforeseen challenges. VigilNet [11] is designed to support long-term military surveillance using a sensor network consisting of 200 nodes and covering 100×100 square meters. ExScal [2] is an attempt to deploy a sensor network at "extreme" scale. The system consists of about 1,000 sensor nodes and 200 backbone nodes, covering $1{,}300 \times 300$ square meters. Dutta et al. [7] report a network deployment Trio of 557 solar-powered motes for multitarget tracking. SenseScope [3], [4] is a real-world deployment that took place on a rock glacier, consisting of about 100 sensor nodes.

Most of above mentioned systems are not clearly proper for network measurement due to the following two reasons. First, those systems are organized hierarchically. Usually sensor nodes reside in the lowest tier and perform sensing functions; a number of gateways, acting as cluster heads at the second tier, collect data traffic within their clusters and forward data to a central server through another communication channel. Such hierarchical architecture inherently alleviates the negative impact induced by large-scales, thus hardly reflecting the performance of general and homogeneous ad hoc sensor networks. Second, no single system has integrated large-scale (e.g., hundreds of nodes) and long-lifetime (e.g., one year) into a cohesive whole. In other words, those deployments have achieved either scale or lifetime, but usually not both.

In contrast, in this paper, we conduct experiments on GreenOrbs, a large-scale, long-term, general-purpose, and homogeneous environment monitoring system. The measurement results of GreenOrbs provide us deep understanding into deployment methodology of sensor networks at large-scales.

In the context of wireless sensor networks, a number of empirical studies present network measurement results.

Link quality is one of the most important indicators for wireless communication and thus attracts many research efforts. Srinivasan et al. [20] conclude from measurements on MicaZ motes with CC2420 radios that RSSI is a good estimate of link quality. Zuniga and Krishnamachari [28] study the transition region and quantify its influence. Studies such as [5], [19] emphasize the temporal performance dynamics of wireless links and provide important findings about such phenomenon. More details about measurements in wireless sensor networks can be found in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2012.216.

The results presented in those empirical studies are basically obtained from testbeds of tens of sensors. In contrast, the measurements of GreenOrbs are fairly comprehensive, from low-level radio signal strength and link quality to high-level routing and data traffic issues.

3 GreenOrbs and System Implementation

In this section, we present the overview of GreenOrbs system. In the process, we first introduce the system background and the forestry applications of GreenOrbs. Second, we explain the building blocks of GreenOrbs, including the hardware, software, and protocols. The evolutional process of GreenOrbs deployments is also described.

3.1 Building Blocks

Hardware. GreenOrbs employs the TelosB mote with a MSP430 processor and CC2420 transceiver. On a sensor node, the program flash memory is 48K bytes. The measurement serial flash is 1,024K bytes. The RAM is 10K bytes. More details about the sensor nodes are introduced in Appendix B and C, which are available in the online supplemental material.

Software and Protocols. To enable sustainable sensing in the forest, in the early GreenOrbs deployments, we used to adopt the Low Power Listening (LPL) interface to enable LPL on the duty-cycled nodes. Later, we realized that the forestry applications usually require synchronous sensor reading, so we modified the synchronized duty cycling mechanism. Based on the network synchronization, every hour all the nodes wake up simultaneously, keep radio on for the same time period, and then switch to sleep simultaneously.

The software on the GreenOrbs nodes is developed on the basis of TinyOS 2.1. The main data stream is multi-hop data collection from the ordinary nodes to the sink. The *Data Collector* component based on CTP [9] is employed for this purpose. Hence *Configurator* component based on Drip [22] is devised to achieve efficient data disseminations. Meanwhile, the FTSP protocol [16] plays the functions of network-wide synchronization, so as to enable the globally synchronized duty cycles. The detailed software components are illustrated in Appendix D, which is available in the online supplemental material.

3.2 Evolution and Deployments

The first GreenOrbs deployment was carried out in July 2008. Ever since then, GreenOrbs has experienced a number of deployments at different places, with different scales, and for different durations. The detailed evolution of the deployment is introduced in Appendix E, which is available in the online supplemental material.

4 Measurement Methodology

4.1 System Settings

In the measurement period of GreenOrbs, we decrease the beaconing frequencies of CTP and FTSP to reduce the control overhead. The CTP protocol has been further tailored to the GreenOrbs system. We have modified CTP and made the routing mechanism less sensitive to link failures. An update of the routing table is triggered only when the number of retransmissions exceeds a predefined threshold.

As described in Section 3.1, the Drip-based *Configurator* enables us to regulate the operational parameters of nodes without collecting them back. Drip uses TrickleTimer [12], and when receiving a new version of data, it disseminates data at the minimum interval. Otherwise, it doubles the broadcasting interval until the maximum value. We set the maximum interval of TrickleTimer as 20 seconds. Appendix F, which is available in the online supplemental material, lists the details of the configurable parameters.

4.2 Data Set

The data set used for analysis, evaluation, and experiments in this paper mainly comes from the operational period of GreenOrbs in December 2009. It contains data of 29 consecutive days, counts 2,540,000 data packets. To conduct comprehensive observation on the large-scale sensor network system, during the above mentioned period we have regulated the nodes with different combinations of operational parameters. The detailed configurations of the data set are shown in Table 1. We were in a progress of gradually enlarging the network. During the measurement study, we vary the core system parameters like transceiver power level, data rate, and duty cycling rate so as to investigate the system performance with various conditions.

TABLE 1
Configurations Used in the Data Set

Trace No.	Network Scale	Power level	Data Rate (packets/hour)	Duration (hour)	Duty cycle
1	100	15	3	60	No
2	200	15	3	25	No
3	330	15	3	300	No
4	330	15	12	24	No
5	330	15	18	100	No
6	330	15	27	30	No
7	330	15	54	3	No
8	330	15	108	3	No
9	330	31	12	1	No
10	330	21	12	1	No
11	330	15	12	1	No
12	330	8	12	1	No
13	330	15	3	150	8%
14	330	15	60	12	8%

4.2.1 Back End Data Set

The back end data set refers to the entire data set collected at the sink via multihop routing, denoted by D_{sink}. D_{sink} is made up of three categories of traces.

Routing trace, denoted by $T_{routing}$ and encapsulated as packet of type 41. It mainly records the routing path of a packet, namely the sequence of relaying nodes between the source and the sink. The sensor readings, such as temperature, humidity, illuminance, and carbon dioxide, are included in $T_{routing}$ as well.

Link trace, denoted by T_{link} and encapsulated as packet of type 42. It includes the list of neighbor node IDs. For each neighbor node, the RSSI, LQI, and Estimated Transmission Counts (ETX) [6] are included in T_{link} as well.

Node statistics trace, denoted by T_{stats} and encapsulated as packet of type 45. T_{stats} is a large set of statistical information on each node, including the cumulative time of radio power on, the cumulative number of transmitted and received packets, the cumulative number of packet drops (due to receive pool overflow, d transmit queue overflow, and transmit timeout), the cumulative number of transmissions that are not ACKed, retransmissions, received duplicate packets, and the parent changes with the CTP.

4.2.2 Out-Band Measurements

Due to the packet losses and various failures in wireless sensor networks, the back end data set is far from sufficient for characterizing the GreenOrbs system at a full scale. Thus, we introduce three out-band measurement techniques, namely overhearing, beaconing, and local logging.

Overhearing. We deploy multiple sniffers in the network to overhear the network traffic. A sniffer is a TelosB mote, which passively listens without sending out any packets. In our early attempt we let the sniffers store all the overheard data in their serial flash. The 1M bytes flash on TelosB mote was soon found too limited for durative overhearing, so we connect the sniffers to stable and powerful devices, for example, a laptop, to record all the overheard data. The locations of sniffers are carefully selected so that the combined communication ranges of the sniffers cover the entire network. The data from sniffers are denoted by $D_{sniffer}$.

Beaconing. In many scenarios, we find a number of nodes never successfully report data to the sink, making us fail to find out the cause by using D_{sink} only. Therefore, in some of the experiments, we let each node actively broadcast beacons periodically. The content of the beacon is similar to that in T_{stats} (packet type 45). The broadcast beacons are overheard by the nearby sniffers and stored in $D_{sniffer}$. The neighbor nodes heard the beacon from a node can also use it to update T_{link}.

Local logging. Other than the networking information, the fine-grained local events on the nodes are equally important for us to understand their behavior and interactions. As a necessary complement, every node locally logs events such as transmissions, retransmissions, ACKs of packet receptions. Each event is recorded with six bytes, where two bytes denote the event type and the other four bytes denote the timestamp of an event. The data set of local logging is denoted by D_{log}. Since the deployment is still in operation, we do not collect all the nodes back to read their logs. D_{log} is currently used as a backup data set for diagnosis on some faulty nodes.

4.3 Measures and Derivations

4.3.1 Primary Measures

Yield. We use *yield* [25] to measure the quantity of the collected data. The *network yield* measures the quantity of the entire network while *node yield* measures the quantity of an individual node. Specifically, the *node yield* is calculated by

$$Yield_i = \frac{\text{\# of data pkts received by the sink from } i \text{ during } w}{\text{\# of data pkts sent by } i \text{ during } w},$$

where i is the node ID, and w is a measurement period. The *network yield* is calculated by

$$Yield = \frac{\text{\# of data pkts received by the sink during } w}{\text{\# of data pkts sent by all nodes during } w}.$$

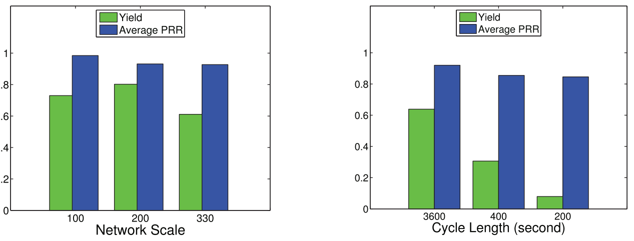
Packet Reception Ratio (PRR)/Loss Ratio. We use *PRR* to measure the quality of a link. Throughout this paper, we use two-way link *PRR*, i.e., we consider a successful transmission only if the sender receives an ACK

$$PRR = \frac{\text{\# of ACKed data pkts}}{\text{\# of sent data pkts}}.$$

The packet loss ratio is $PLR = 1 - PRR$. We show the details of calculating PRR for our data in Appendix G, which is available in the online supplemental material.

Packet Delivery Ratio (*PDR*). *PDR* is defined as the ratio of the amount of packets received by the destination to those sent by the source. Since the transmissions are reinforced with retransmissions, *PDR* can be higher than link *PRR* in practice.

End-to-end delay. The *end-to-end delay* of a packet is the time difference between the sending time at the source node and the reception time at the sink. We stamp each data packets when it is first transmitted from the source node and when it is received at the sink. The FTSP protocol is used to ensure time synchronization.



(a) (b)

Fig. 2. *Network yield* and *PRR* v.s. (a) different network scales, and (b) different traffic loads.

Correlation Coefficient. Correlation coefficient is a statistical measure of association between two variables, for example, the ETX value and the PDR. The range of correlation coefficient is $[-1, 1]$. The sign denotes whether two variables are positively or negatively related and the absolute value corresponds to their correlation strength. For example, the correlation coefficient equals to 1 when two variables are in positive linear relationship, -1 in the case of a negative linear relationship, and 0 when two variables are completely independent. More details about the correlation can be found in Appendix H, which is available in the online supplemental material.

5 Basic Observations

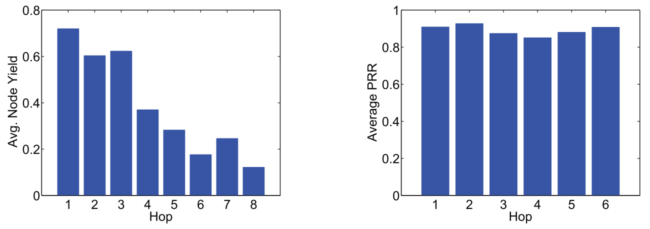
In this section, we present a set of basic observations on the operation of the system. Our observations range from the high-level system performance down to the detailed behaviors at the link level. From those basic observations, we summarize the network characteristics and explore the reasons that bottleneck the performance when the network scales.

5.1 Network Characteristics

The *network yield*, the ratio of packets successfully received at the sink side to the total number of packets generated by all the nodes, is a primary metric that evaluates the system performance. It provides us a global indication on how complete the network-wide data are collected. Another metric is link PRR that estimates the percentage of successfully ACKed packets over all the transmissions plus retransmissions, giving us a microscopic indication on how the transmissions perform on the links.

Fig. 2a exhibits the system performance when the network scales from 100 nodes to 200 nodes, and then to 330 nodes. During the measurement, the data generation rate at each node is three packets per hour. There is not apparent trend of changes on the *network yield*, partially because the traffic inserted into the network is relatively low. On the other hand, the average link *PRR* across all the links does not exhibit apparent difference when the network scales.

We then measure the same metrics while exerting different traffic load over the network, keeping the network scale as 330. Letting each node generate three packets per cycle, we increase the traffic load in a stepwise manner by shortening the cycle lengths, namely 3,600, 400, and 200 seconds. As Fig. 2b shows, the increasing traffic load severely degrades the system performance. As depicted in Fig. 2b, the *network yield* rapidly drops from over 60 percent to less than 10 percent.



(a) Average node yield (b) Link PRR

Fig. 3. System performance for different categories of nodes.

A natural question raised from the above observation is: Whether the degradation in terms of *network yield* is due to the throughput bottleneck around the sink? Indeed, the "hot area" around the sink has recently been widely reported in a number of literatures. The research communities also propose a variety of protocols to mitigate such a problem [17], [27]. However, if we carefully analyze the data provided by Fig. 2b, we notice that the highest network throughput occurs when the cycle length is set at 400 seconds. The average packet size in GreenOrbs is 100 bytes. The goodput of data reception from the network can be calculated by: $3 \times 330\, packets \times 100\, bytes \times 8 \times 31\%/400\mathrm{s} = 0.61$ Kbps. Such a goodput is far less than that reported in [9] (at least 20 packets/second). This huge gap clearly suggests that the network is far from being bottlenecked before the sink bandwidth is used up. Hence, the follow-up question is: Now that the area around the sink receives relatively high traffic load and severe transmission contentions, is it the place where a large portion of the packet losses occur?

In Fig. 3, we present a close look at the system behaviors using Trace No. 6. We categorize the sensor nodes in the network according to their hop counts to the sink. Note that a node sometimes switches its parent, resulting in dynamic routing paths to the sink of different hop counts. In the statistics, we use a precise granularity to categorize the nodes with such behavior. The packets sent from the same node with different hop counts are separately counted into different categories. Fig. 3a depicts the PRR from the nodes of different hop distances to the sink. There is a clear trend that the nodes farther from the sink have a lower *PDR* to the sink. Nevertheless, Fig. 3b depicts the link *PRR* according to links' hop distances to the sink. There are apparent differences among all the links. This is direct evidence, which reveals that the area around the sink is not the rendezvous of packet losses. Otherwise, the PRRs of different categories of nodes should not deviate in the manner of Fig. 3a.

To investigate the cause of packet losses, we further classify the packet losses into three categories:

- Transmit timeout: the packet is (re)transmitted 30 times and dropped due to not receiving the ACK signal. Such packet drops are mainly due to the poor quality of the wireless channels or severe collisions during wireless transmission.
- Receive pool overflow: the packet is successfully received at the receiver end but immediately

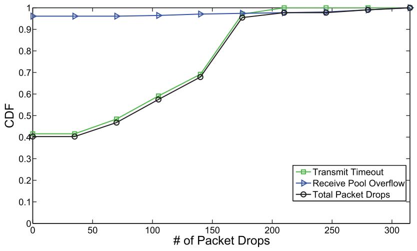


Fig. 4. CDF of the nodes with different numbers of packet drop occurrences.

dropped due to the forwarding queue overflow. This type of packet drop is mainly caused by the excessively heavy data congestion at the receiver.

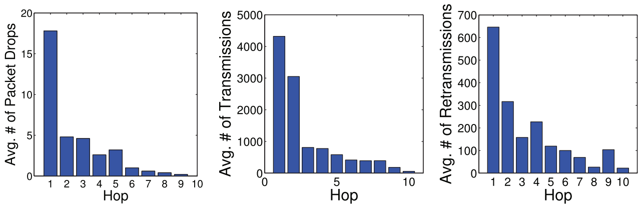
- Send queue overflow: the packet fails to be inserted into the forwarding queue, mainly due to the mismatch between sensor processing capability and the high rate of packet arrival.

We examine all packet losses in Trace No. 5. Among all packet losses, Transmit Timeout accounts for 61.08 percent and Receive Pool Overflow accounts for the rest 38.92 percent. No Send Queue Overflow is detected.

We further investigate the distribution of packet drop occurrences among the nodes, as shown in Fig. 4. The packet drops due to Transmit Timeout are evenly distributed across different intensities. Nearly 90 percent nodes have less than 20 Transmit Timeout losses and no node has more than 50 Transmit Timeout losses. Surprisingly, we find that over 95 percent nodes do not have any Receive Pool Overflow drop. All the Receive Pool Overflow drops (38.92 percent of all packet drops) occur on less than 5 percent nodes. Such a finding implies that there exist a very small portion of nodes in the network which play critical roles, taking excessively high traffic load, and responsible for the major portion of packet losses. Those nodes should be carefully considered in practical network protocol design.

5.2 Investigating Critical Nodes/Links

We take a deep look into the network and investigate the node level behavior. We reorganize our observations and exhibit the individual node performance according to their hop distances to the sink, as shown in Fig. 6. An intuitive impression is that the nodes near the sink take more traffic load and hence have apparently poorer performance. However, we still cannot conclude that the critical nodes mainly lie near the sink, as Fig. 6 only gives us the aggregated performance of many nodes. More details results on node performance for different traffic pressure is provided in Appendix I, which is available in the online supplemental material.



(a) Packet drops (b) Traffic load (c) Retransmissions

Fig. 6. Node performance of different categories.

Those critical nodes need to be individually identified within the network. For this purpose, Fig. 5 plots all the 330 nodes. In total, eight snapshots of eight consecutive operational periods are included. Each node is colored according to the traffic load it takes. A deeper color indicates higher incoming traffic load at a node. It can be seen that those critical nodes take excessive traffic. Further investigation shows that some critical nodes reside on the area with more dynamic environment, for example, near the road. Those nodes are distributed across the entire deployment area instead of concentrated near the sink (the black node in the figures). In Appendix K, which is available in the online supplemental material, we index the nodes according to their traffic load and find that less than 10 percent critical nodes commit 80 percent traffic load and thus 61.06 percent of the packet loss. They act as bottlenecks of the system. This further suggests that such a set of critical nodes are relatively stable.

5.3 Looking into the Links

As our network-wide statistics suggest, there exist a small portion of critical nodes that bottleneck the performance of the entire network. According to our statistics on different categories of the packet drops, both Transmit Timeout and Receive Pool Overflow contribute a large portion, implying that both congestion and link losses are possible causes that degrade the network performance. We are interested in the reason behind such a phenomenon. A question yet we want to answer is whether the existence of such critical nodes is mainly due to the poor quality of wireless communication, severe congestion or contention accompanied with the

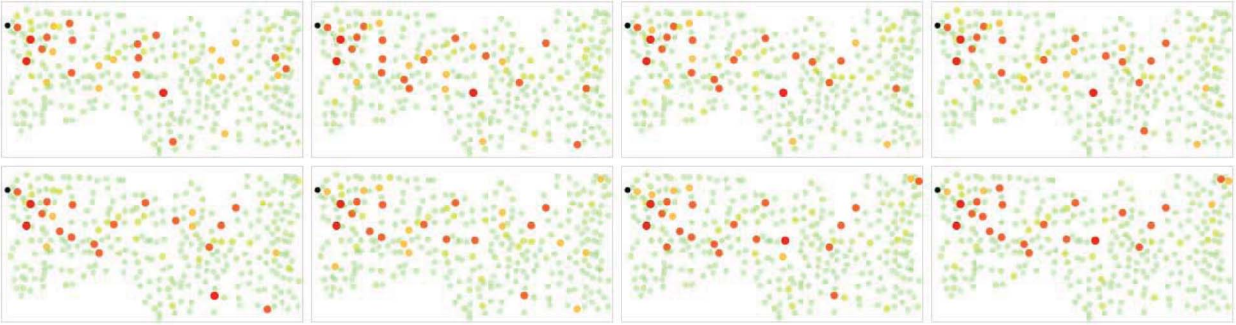
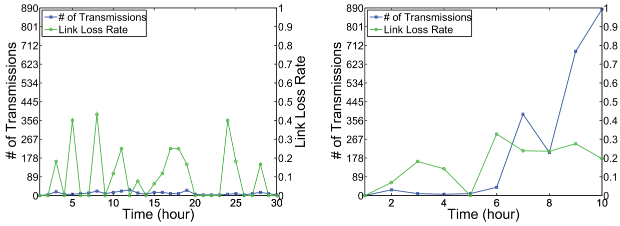


Fig. 5. The traffic distribution over the network.



(a)

(b)

Fig. 7. The traffic and PRR on two typical links.

unbalanced traffic overhead. To answer this question, we further take a look into the link behavior.

Fig. 7 shows the observation on two typical links. We find that the link loss rate fluctuates with time and it seems independent from the traffic load. An immediate guess is that such link dynamics may come from the environmental dynamics. Recall that our system is indeed deployed in the wild. To further explore the link loss fluctuation, we adjust the transmission power of the nodes. Intuitively, as the transmission power is increased, the received signal strength will be strengthened and the link *PRR* will be improved. The level of transmission power is set at 8, 15, 21, and 31 (Traces No. 9-12), respectively. In CC2420, they correspond to the sending power of slightly above -15 dBm, -7 dBm, around -4 dBm, and near 0 dBm. More results on the impact of power setting to delay performance can be found in Appendix J, which is available in the online supplemental material.

With such observations we have to carefully reconsider the way we used to view the wireless links in sensor networks. Are they inherently unpredictable with fluctuating quality? If so, are the link fluctuations due to the unpredictable environmental dynamics? Otherwise, assuming the wireless links as indeed good medium for data communications, do the current designs and protocols simply fail to make the best use of them?

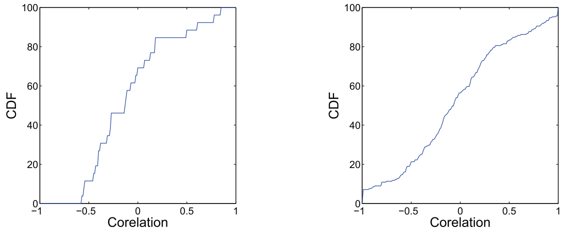
6 Who Moved Our Cheese?

As we have experienced from our basic observations, the network cannot unlimitedly scale due to the physical resource constraint. In this section, we summarize from our basic observations and try to explore the major reasons that limit the system scale. What is the dominant resource that is at the first depleted when the network workload scales? Are such resource balanced used? Where are the places of resource depletion that bottleneck the entire network? How should existing protocols be improved to adapt to large-scale sensor network characteristics? With those questions, we proactively look into our data trace and conduct a new set of experiments.

6.1 The Last Straw that Breaks the Camel's Back

As previously shown in Section 5, when the size of the network scales and the traffic load increases, the overall system performance drops, especially after the scale exceeds a limit.

Differing previous studies, our measurement results suggest that the "hot area" problem around the sink does not seem to play a major role in degrading the performance of our system. Instead, we observe a set of critical nodes that are distributed across the network, receiving excessively high-traffic input, with fluctuating link loss rate, and accounting for a large portion of packet drops. Current routing protocols do not emphasize on those cases. As a result, the routing structure may overreacts to path dynamics, leading the network traffic concentrating from one area to another, creating "hot" spots from time to time. Meanwhile, some nodes may reside at an important position, absorbing a large amount of traffic. For example, nodes near the ridge need to relay traffic for nodes from one side to the other. This should be the same for different network densities. We need to complement current dynamic routing protocols, like CTP used in our settings, to successfully handle those cases in time.



(a) ETX v.s. PDR

(b) ETX v.s. Delay

Fig. 8. The Pearson Correlation between (a) ETX and PDR on the path, and (b) ETX measure and packet delivery deday.

We examine the ETX value stored in each sensor node for routing selection and compare it with the real path quality we measure from the packet delivery, including the *PDR* and the end-to-end delay along the path. We use Pearson product-moment correlation coefficient to measure the correlation between the ETX value used for routing selection and the path *PDR* and end-to-end delay that reflects the real performance of the selected routing path. Fig. 8 depicts the CDF of the correlation coefficient. We find that the coefficient almost exhibits a random effect between -1 and 1, indicating that the ETX indicator hardly reflects the real path quality for most of the time in the large-scale network. This is because there are retransmissions for each hop and even when the ETX is large, the PDR can still be high.

In particular, besides ETX, most currently used link-based path indicators like RSSI or LQI aggregate, focus more on the quality of data transmissions on the links, but overlook the quality of data forwarding inside the nodes. In Fig. 9, we post a 20 hour statistics on the data forwarding behaviors of a particular node (node 225). In the first half, it gives a satisfactory performance, forwarding almost all of the input data packets successfully. At some intermediate time spot around 10 hours, this node happens to drop all the input data packets while still successfully sending its own data packet out. This weird behavior may relate to a program bug that leads to locked memory of the forwarding queue in CTP with special concurrent operations. The real problem is that, even when such a node drops all incoming data packets it receives, it is still consistently selected as the parent in the routing tables of many nodes for the rest of time. Such a phenomenon is largely due to the fact that the ETX indicator does not reflect the data drops

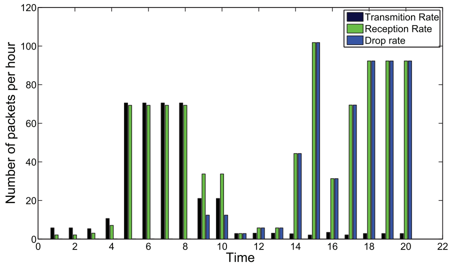


Fig. 9. The packet reception, forwarding, and drop at node 225 within 20 hours.

within the node. The ETX measured at node 225 is always good and broadcasted to its neighboring nodes, consistently absorbing the traffics and dropping them. Against such a problem, an aggregated indicator is urged reflecting both link transmission quality and intranode forwarding quality.

Thus our first conjecture is that: the dominating bottleneck of a large-scale sensor network does not only come from the "hot area" around the sink. It is very possible that some of the intermediate nodes bottleneck the entire network while existing widely used indicators may not accurately capture them.

6.2 How Dynamic Is the Environment?

According to our observations, the packets drops on links contribute the largest part of packet drops, although it does not exhibit as strong concentration as the overflow drops do.

Our investigation shows that the link loss rate fluctuates with time and an immediate guess is such link dynamics may come from the dynamic environment. In fact, many existing works have reported the possibility of such dynamics from environment. We conduct an independent set of experiments in our deployment area. We place two sensor nodes in the same environment as where our system is deployed and measure the link quality under different settings. The two nodes are placed 20 m, and 50 m apart separately. We let one node send data packets and the other receive. We measure the RSSI and link loss rate at the receiver.

As the result in Fig. 10 suggests, the RSSI, which is a major indicator that measures the quality of propagated signal is relatively stable through most of the time and settings. The fluctuation of RSSI is almost composed of a series of sparkling burrs. That is quite possible from the interference from nearby 802.11b AP signals, as reported in [19]. Only in the experiment setting of 50 m distance and 20 Hz sending rate, the RSSI varies around −90 dBm, and there appear some observable link loss. This is mainly because the CC2420 transceiver has a hardware sensitivity threshold to the incoming signal strength at around −93 to −87 dBm [1]. More details of the experiment can be found in Appendix M, which is available in the online supplemental material.

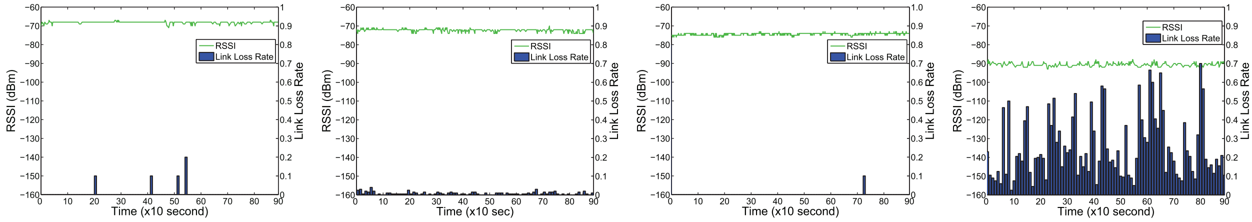
Above measurement results give us a direct implication that the signal propagation in the wild is not as dynamic as what we assumed. Recall that in our observations on the link performance within our system, the link loss rate fluctuates far more intensively (see Fig. 7) and there is not an apparent correlation between the link loss rate and the traffic overhead on that link. However, as our network-wide statistics suggest, the high-link loss usually occur at those nodes within or near high-traffic regions. Such observations suggest us a quite real possibility that the fluctuating link loss is due to the collisions of concurrently transmitted packets in those regions. Such packet contentions are not detected by the CSMA mechanism, resulting in improper concurrency. Considering the high density of node deployment in the sensor network, the traditional wireless "hidden terminal" problem might be far more popular than else where like 802.11 AP network or MANET where the network is usually at a small density.

Thus, our second conjecture is that: most of the wireless links used in our network are physically stable. The dynamics of sensor networks do not only come from the external environment but also the internal network operations. The inherent concurrency of operations among different nodes should be further investigated and considered in designing scalable network protocols.

6.3 Adaptive Routing Design

While our measurement results reflect that the environment introduces very limited dynamics to the network, the impact from the deployment environment itself is substantial.

When deploying the sensor nodes we try to place them uniformly across the field, aiming to provide a uniform networking structure. As explicitly shown in Fig. 1, however, the uniform sensor deployment does not yield a uniform networking overlay. Some nodes have excessively high-neighbor degree and forward times of data than others. Some of them become critical nodes later during our test, bottlenecking the network performance. We fail to achieve logical uniformity from geological uniformity, largely due to the inherent irregularity of the deployment environment. The bumpy floor in the wild, woods standing in between, slope of the hill, and so on, all environment factors make the signal propagation irregular, resulting in the overlay construction unpredictable before the system is really deployed. Only after the overlay characteristics are



(a) 1Hz and 20m

(b) 20Hz and 20m

(c) 1Hz and 50m

(d) 20Hz and 50m

Fig. 10. The RSSI and link loss rate measured at an independent pair of nodes.

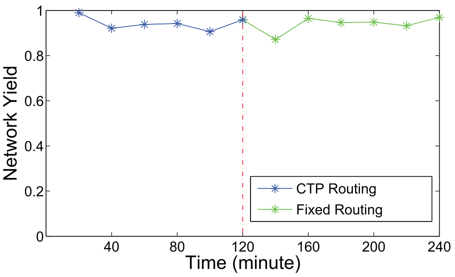


Fig. 11. The performance comparison between original CTP routing and that when the routing table is fixed.

thoroughly studied after deployment, we are able to provide customized schedule in the routing layer that optimizes the system performance.

While current dynamic routing approaches (like CTP used in our system) aim to adjust according to the network dynamics, they usually lack tailored optimization in adapting to the surrounding environment. Besides, according to our observations, the environment impacts are relatively stable, providing us adequate room in designing comparatively static while highly optimized routing protocols.

Fig. 11 exhibits our initial attempt in proving our argument. During the system operation, we let the network run with CTP routing for 120 minutes and then fix the routing tables for another 120 minutes letting each node forward the passing by packets to a fixed parent node. According to the measurement results, there is no apparent difference on the performance of *network yield* in the two working periods for the relative stable environment.

We believe, with careful consideration on the actual network overlay under the practical environment and an intelligent learning process, it is very possible that a highly optimized static routing structure outperforms existing dynamic routing approaches when used in a large scale sensor network. The static routing structure can be made adaptive to the environment changes on an event triggered basis. The routing structure will only be reconstructed when sharp events happen like intensive weather changes, large relief variations, a broad area of sensor damages, and so on, and after adequate knowledge about the new environment is learned.

Thus, our third conjecture is that: The environment, while with less dynamics than we expected in our network, has an unpredictable impact on the sensor network system running under it. We can improve current dynamic routing approaches by learning its unique characteristics. We suggest that an event-based static routing structure may have better performance in operating a large-scale sensor network in the wild environment.

7 Conclusions

In this paper, we conduct a measurement study on a large-scale operating sensor network system, GreenOrbs, with up to 330 nodes deployed in the wild. We aim to comprehensively understand how the sensor network performs when it scales to contain hundreds or even thousands of nodes. We instrument such an operating network throughout the protocol stack. The contribution of this work is twofold.

First, to the best of our knowledge, we are the first to conduct a long term and large-scale measurement study on an operating sensor network in the wild. We present observations across a variety of layers in the network that provide research community empirical experiences on how practical problems affect when the sensor network scales.

Second, based on our basic findings from the system measurement, we further propose and initially attempt to validate three conjectures that provide guidelines for future algorithm and protocol designs with larger scale sensor networks. We also discuss the generality of our observations in Appendix N, which is available in the online supplemental material. In summary, 1) we think it might be very possible that some of the intermediate nodes bottleneck the entire network, and most of currently used indicators may not accurately capture them; 2) most of the wireless links in large scale sensor networks are physically stable. The dynamics mainly come from the inherent concurrency of network operations which should be further investigated and considered in designing scalable network protocols; (3) the environment, while with insignificant dynamics, has an unpredictable impact on the sensor network under it. We suggest that an event based routing structure can be trained optimized and thus better adapt to the wild environment when building a large-scale sensor network.

Acknowledgements

This study is supported in part by the NSF China Major Program 61190110, National High-Tech R&D Program of China (863) under grant No. 2011AA010100, China 973 under grant No. 2011cb302705, the NSFC Distinguished Young Scholars Program under Grant 61125202, NSFC under Grants 61170213, 61103187, 61228202, and 61202359, China Postdoctoral Science Foundation under Grant 2011M500019 and 2011M500330, Singapore MOE AcRF Tier 2 grant MOE2012-T2-1-070, NSF CNS-1035894, and NSF ECCS-1247944. A preliminary work has been presented in IEEE INFOCOM 2011 [14].

References

[1] CC2420 data sheet: http://focus.ti.com/lit/ds/symlink/cc2420.pdf, 2013.

[2] A. Arora, R. Ramnath, and E. Ertin, "ExScal: Elements of an Extreme Scale Wireless Sensor Network," *Proc. IEEE 11th Int'l Conf. Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2005.

[3] G. Barrenetxea, F. Ingelrest, G. Schaefer, and M. Vetterli, "The Hitchhiker's Guide to Successful Wireless Sensor Network Deployments," *Proc. ACM Sixth Conf. Embedded Network Sensor Systems (SENSYS)*, 2008.

[4] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "SensorScope: Out-of-the-Box Environmental Monitoring," *Proc. Int'l Conf. Information Processing in Sensor Networks (IPSN)*, 2008.

[5] A. Cerpa, J.L. Wong, M. Potkonjak, and D. Estrin, "Temporal Properties of Low-Power Wireless Links: Modeling and Implications on Multi-Hop Routing," *Proc. ACM Sixth Int'l Symp. Mobile Ad Hoc Networking and Computing (MOBIHOC)*, 2005.

[6] D.S.J.D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A High-Throughput Path Metric for Multi-Hop Wireless Routing," *Proc. MobiCom*, 2003.

[7] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Whitehouse, and D. Culler, "Trio: Enabling Sustainable and Scalable Outdoor Wireless Sensor Network Deployments," *Proc. Fifth Int'l Conf. Information Processing in Sensor Networks (IPSN)*, 2006.

[8] E. Ertin, A. Arora, R. Ramnath, M. Nesterenko, V. Naik, S. Bapat, V. Kulathumani, M. Sridharan, H. Zhang, and H. Cao, "Kansei: A Testbed for Sensing at Scale," *Proc. Fifth Int'l Conf. Information Processing in Sensor Networks (IPSN),* 2006.

[9] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection Tree Protocol," *Proc. ACM Seventh Conf. Embedded Networked Sensor Systems (SENSYS),* 2009.

[10] T. He, P. Vicaire, T. Yan, Q. Cao, G. Zhou, L. Gu, L. Luo, R. Stoleru, J.A. Stankovic, and T.F. Abdelzaher, "Achieving Long-Term Surveillance in VigilNet," *Proc. INFOCOM,* 2006.

[11] T. He, P. Vicaire, T. Yan, Q. Cao, G. Zhou, L. Gu, L. Luo, R. Stoleru, J.A. Stankovic, and T.F. Abdelzaher, "Achieving Long-Term Surveillance in VigilNet" *ACM Trans. Sensor Networks (TOSN),* vol. 5, no. 1, pp. 1-39, 2009.

[12] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks," *Proc. First Conf. Symp. Networked Systems Design and Implementation (NSDI),* 2004.

[13] M. Li and Y. Liu, "Underground Coal Mine Monitoring with Wireless Sensor Networks," *ACM Trans. Sensor Networks (TOSN),* vol. 5, no. 2, pp. 1-29, 2009.

[14] Y. Liu, Y. He, M. Li, J. Wang, K. Liu, L. Mo, W. Dong, Z. Yang, M. Xi, J. Zhao, and X.-Y. Li, "Does Wireless Sensor Network Scale? a Measurement Study on Greenorbs," *Proc. IEEE INFOCOM,* 2011.

[15] Y. Liu, Q. Zhang, and L.M. Ni, "Opportunity-Based Topology Control in Wireless Sensor Networks," *IEEE Trans. Parallel and Distributed Systems (TPDS),* vol. 21, no. 3, pp. 405-416, Mar. 2010.

[16] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "FTSP: The Flooding Time Synchronization Protocol," *Proc. Second Int'l Conf. Embedded Networked Sensor Systems (SENSYS),* 2004.

[17] S. Olariu and I. Stojmenovic, "Design Guidelines for Maximizing Lifetime and Avoiding Energy Holes in Sensor Networks with Uniform Distribution and Uniform Reporting," *Proc. INFOCOM,* 2006.

[18] K. Sakai, S.C.-H. Huang, W.-S. Ku, M.-T. Sun, and X. Cheng, "Timer-Based Cds Construction in Wireless Ad Hoc Networks," *IEEE Trans. Mobile Computing (TMC),* vol. 10, no. 10, pp. 1388-1402, Oct. 2011.

[19] K. Srinivasan, P. Duttaxyd, A. Tavakoli, and P. Levis, "Understanding the Causes of Packet Delivery Success and Failure in Dense Wireless Sensor Networks," *technical report, Stanford Univ. and UC Berkeley,* 2006.

[20] K. Srinivasan and P. Levis, "RSSI is under Appreciated," *Proc. Third Workshop Embedded Networked Sensors (EmNets),* 2006.

[21] H. Tan, I. Korpeoglu, and I. Stojmenovic, "Computing Localized Power Efficient Data Aggregation Trees for Sensor Networks," *IEEE Trans. Parallel and Distributed Systems (TPDS),* vol. 22, no. 3, pp. 489-500, Mar. 2011.

[22] G. Tolle and D. Culler, "Design of an Application-Cooperative Management System for Wireless Sensor Networks," *Proc. Second European Workshop Wireless Sensor Networks (EWSN),* 2005.

[23] 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," *Proc. Third Int'l Conf. Embedded Networked Sensor Systems (SENSYS),* 2005.

[24] X. Wang, Y. Bei, Q. Peng, and L. Fu, "Speed Improves Delay-Capacity Tradeoff in Motioncast," *IEEE Trans. Parallel and Distributed Systems (TPDS),* vol. 22, no. 5, pp. 729-742, May 2011.

[25] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and Yield in a Volcano Monitoring Sensor Network," *Proc. Seventh Symp. Operating Systems Design and Implementation (OSDI),* 2006.

[26] G. Werner-Allen, P. Swieskowski, and M. Welsh, "MoteLab: A Wireless Sensor Network Testbed," *Proc. Fourth Int'l Symp. Information Processing in Sensor Networks (IPSN),* 2005.

[27] X. Wu, G. Chen, and S.K. Das, "Avoiding Energy Holes in Wireless Sensor Networks with Nonuniform Node Distribution," *IEEE Trans. Parallel and Distributed Systems (TPDS),* vol. 19, no. 5, pp. 710-720, May 2008.

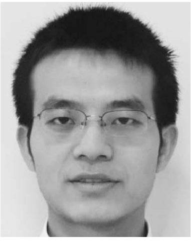
[28] N. Xu, S. Rangwala, K.K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A Wireless Sensor Network For Structural Monitoring," *Proc. Second Int'l Conf. Embedded Networked Sensor Systems (SENSYS),* 2004.

[29] M. Zuniga and B. Krishnamachari, "Analyzing the Transitional Region in Low Power Wireless Links," *Proc. IEEE Comm. Soc. Conf. Sensor, Mesh and Ad Hoc Comm. and Networks (SECON),* 2004.

Yunhao Liu received the BS degree in automation from Tsinghua University, China, in 1995, the MS and PhD degrees in computer science and engineering from Michigan State University, in 2003 and 2004, respectively. He is currently the Chang Jiang Professor at School of Software and TNLIST, Tsinghua University. His research interests include wireless sensor network, peer-to-peer computing, and pervasive computing. He is a senior member of the IEEE and the IEEE Computer Society.

Yuan He received the BE degree in the University of Science and Technology of China, the ME degree in Institute of Software, Chinese Academy of Sciences, and the PhD degree in Hong Kong University of Science and Technology. His research interests include sensor networks, peer-to-peer computing, and pervasive computing. He is a member of the IEEE, the IEEE Computer Society, and the ACM and Tsinghua National Lab for Information Science and Technology.

Mo Li received the BS degree in the Department of Computer Science and Technology from Tsinghua University, China and the PhD degree in computer science and engineering from the Hong Kong University of Science and Technology, in 2004 and 2010, respectively. He is currently an assistant professor in the School of Computer Engineering, Nanyang Technological University. His research interests include wireless sensor networking, pervasive computing, and peer-to-peer computing. He is a member of the IEEE and the IEEE Computer Society.



Jiliang Wang received the BE degree in the Department of Computer Science from the University of Science and Technology of China and the PhD degree in the Department of Computer Science and Engineering from the Hong Kong University of Science and Technology. He is currently with the School of Software in Tsinghua University. His research interest includes wireless sensor networks, network measurement, and pervasive computing. He is a member of the IEEE and the IEEE Computer Society.

Kebin Liu received the BS degree in the Department of Computer Science from Tongji University, and the MS and PhD degrees in Shanghai Jiaotong University, China. His research interests include wireless sensor networks and peer-to-peer computing. He is a member of the IEEE and the IEEE Computer Society.

Xiangyang Li received the BEng degree in computer science and bachelor degree in business management from Tsinghua University, P.R. China, in 1995, and the MS and PhD degrees in computer science from the University of Illinois, Urbana-Champaign, in 2000 and 2001, respectively. He is currently a professor of computer science at the Illinois Institute of Technology. His research interests include wireless sensor networks, computational geometry, and algorithms, and has published more than 200 papers on these fields. He is an editor of IEEE TPDS, *Networks: An International Journal*, and was a guest editor of several journals, such as *ACM MONET*, *IEEE JSAC*. In 2008, he published a monograph Wireless Ad Hoc and Sensor Networks: Theory and Applications. He is a senior member of the IEEE and the IEEE Computer Society and a member of the ACM.