Measurement and Analysis on the Packet Delivery Performance in A Large Scale Sensor Network

Wei Dong[†], Yunhao Liu[‡§], Yuan He[‡], Tong Zhu[§]
[†]CS College, Zhejiang University;
[‡]TNLIST, Tsinghua University; [§]CSE Dept. HKUST
dongw@zju.edu.cn, {yunhao, he}@greenorbs.com, zhutong@cse.ust.hk

***Abstract*—Understanding the packet delivery performance of a wireless sensor network (WSN) is critical for improving system performance and exploring further development and applications of WSN techniques. In spite of many empirical measurements in the literature, we still lack in-depth understanding on how and to what extent different factors contribute to the overall packet losses with respect to a complete stack of protocols at large scales. Specifically, very little is known about (1) When, where, and under what kind of circumstances packet losses occur. (2) Why packets are lost. As a step towards addressing those issues, we deploy a large-scale WSN and design a measurement system for retrieving important system metrics. We propose MAP, a step-by-step methodology to identify the losses, extract system events, and perform spatial-temporal correlation analysis by employing a carefully examined casual graph. MAP enables us to get a closer look at the root causes of packet losses in a low-power ad-hoc network. This study validates some earlier conjectures on WSNs and reveals some new findings. The quantitative results also shed lights for future large-scale WSN deployments.**

I. Introduction

As an emerging technology that bridges cyber systems and the physical world, wireless sensor networks (WSNs) are envisioned to support numerous unprecedented applications. We have witnessed many research studies, deployments of real systems, and substantive practical applications in recent years.

We are still facing severe challenges in designing scalable, long-lived, and high-performance WSN systems. Some of the difficulties come from the fact that the current understanding of WSN is still limited. Therefore, it is necessary to conduct empirical measurements in real-world WSN systems, so that we can better understand the behaviors of large-scale WSNs and facilitate the design of such systems.

In the past years, many WSN protocols have been reported and shown to be effective in testbed or small-scale networks. On the other hand, it is not uncommon to see that many real deployments often adopt a set of tailored protocols to fulfil the application's requirements. We believe that it is important to understand the performance of some *well-principled* protocols *in combination at a large scale*. We are interested in the question: whether they are reliable enough to facilitate the development of future WSNs.

Many deployments have reported the overall packet delivery performance [1], [2]. Also, many empirical measurement studies show how some specific factors impact the packet delivery performance via controlled experiments [3], [4]. However, we usually do not know how and to what extent different factors contribute to the overall packet losses with a complete stack of protocols at a large scale. Specifically, very little is known about (1) When, where, and under what kind of circumstances packet losses occur. (2) Why packets are lost. Answers to the above questions are critical for improving system performance and exploring further development and applications of WSN techniques.

Understanding the packet delivery performance in an operating WSN is challenging due to the following facts. First, data packets might be lost during multi-hop transmissions and thus data collection is incomplete by nature. It is very difficult to acquire the complete information of the internal status. In-depth understanding requires detailed measurement of networking and system metrics, but basically this is far from affordable for resource and energy constrained sensor nodes. Second, operational efforts to disclose the root causes behind packet losses are insufficient, and few efforts have been validated to be effective at large scales. Third, fine-grained measurements usually demand mechanisms which incur unnegligible operational or capital expenses.

As a step towards addressing those challenges, we deploy GreenOrbs, a large-scale and long-term WSN system in the wild. The network we measure is in continuous operation since Dec. 2010 with nearly 400 nodes. For the sensor node hardware, we use the commonly used TelosB nodes. For the software, we use TinyOS and its radio stack, including the LPL MAC, the CTP collection protocol, and the Drip dissemination protocol.

Based on GreenOrbs, we propose MAP, a practical methodology for Measuring and Analyzing the Performance of a large operating WSN. MAP incorporates a well-designed measurement system for retrieving networking and system metrics. MAP includes three steps for analyzing the packet losses. First, it uses robust algorithms to identify the losses as well as important system events. Second, it carefully tracks the interactions inside the WSN system by means of a causal graph. Third, it examines the temporal-spatial correlations among system events having casual relationships.

Using this methodology, we are able to conduct a deep examination of packet losses. For example, for a recent deployment of our system, Figure 1 shows the spatial distribution of packet losses along with the geometric network topology. The red nodes are those with packet delivery ratio (PDR) less than 90%, and the length of the radius indicates the number of lost packets. We note that while previous reports usually show us the delivery performance as in Figure 1, they usually lack detailed analysis on classifying the losses into smaller categories that can be useful for further analysis.

978-1-4673-5946-7/13/$31.00 ©2013 IEEE

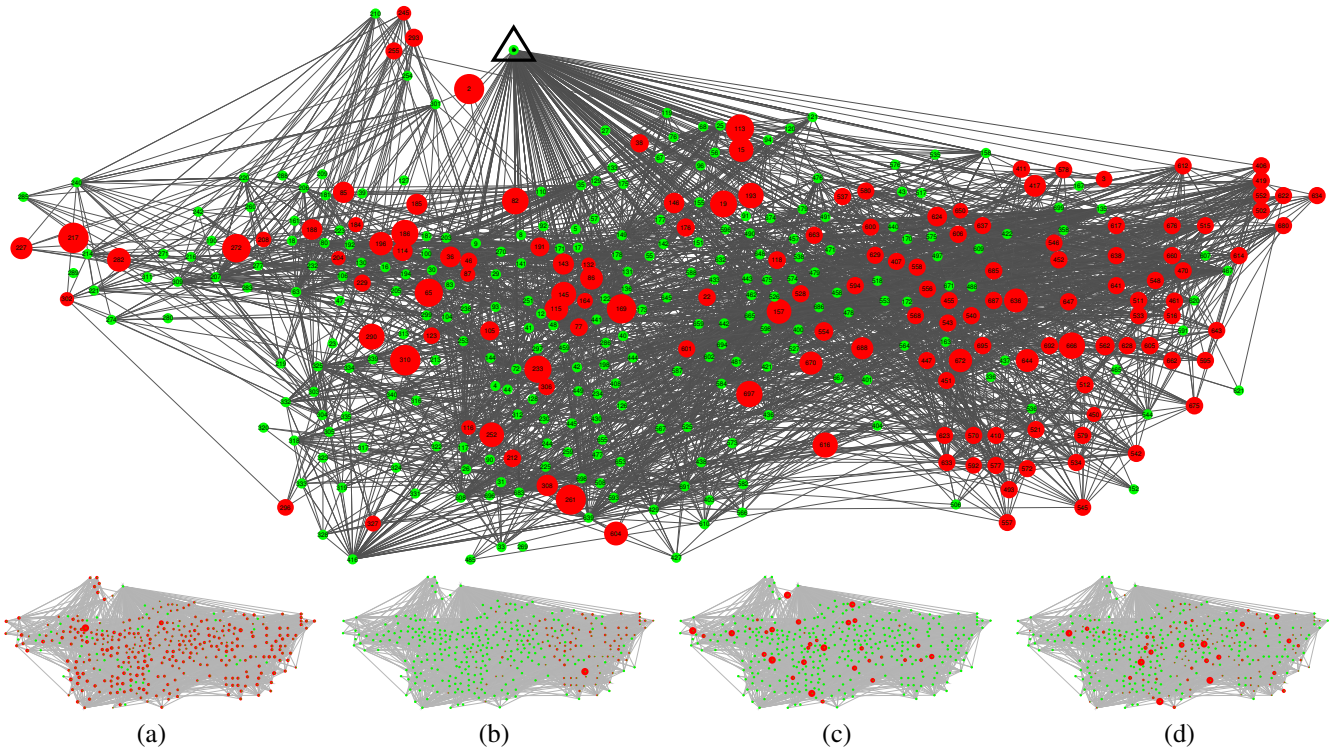


Fig. 1: Network topology where the node with a triangle is the sink node. The green nodes represent nodes with PDR≥90%. The red nodes represent nodes with PDR<90%, and the length of the radius indicates the number of lost packets. The subfigures illustrate the loss distributions of four identified categories: (a) corruption (b) loop overflow drops (c) env-no-ack drops (d) interference-no-ack drops.

The combination of multiple factors makes the overall packet losses exhibit complex patterns that are extremely difficult to reason about. MAP takes a further step to decompose the overall losses into smaller categories that can be related closer to the underlying causes. It can also reveal the spatial-temporal distributions of different losses. For example, Figures 1(a-d) show four major categories of causes. Interestingly, we see different spatial distributions of packet losses caused by different events, with each exhibiting a specific pattern that can better be explained.

The contributions of this study are summarized as follows. (1) We examine the packet delivery performance with a complete stack of TinyOS protocols in a large-scale operating WSN. (2) We develop a methodology to identify the losses as well as investigating the underlying causes for those losses. (3) We quantify to what extent each individual cause contributes to the overall identified losses as well as the loss pattern for each individual category. (4) We give implications and lessons learned to guide future WSN designs and deployments.

The rest of this paper is structured as follows. Section II describes the related work. Section III introduces the network and the particular datasets we use in our study. Section IV shows basic statistics of the network. Section V presents the loss identification algorithm and the spatial-temporal overall distribution of packet losses. Section VI describes the methodology for revealing the root causes. Section VII discusses limitations of our approach. Section VIII summarizes implications and lessons we have learned before we conclude in Section IX with an outlook on future works.

II. Related Work

Many WSN prototypes are deployed in the recent years. During the year 2002–2003, a WSN for habitat monitoring at Great Duck Island [5] is deployed. Tolle et al. [6] report a sensor network consisting of 33 nodes to monitor the microclimate of a redwood tree, covering an area of about 50 square meters. Werner-Allen et al. [7] have deployed a WSN of 16 nodes to monitor an active volcano. Those efforts have provided to the community some basic findings, which act as the early guidance to design the building blocks of modern WSNs. The representativeness of their findings, however, is restricted by the system scale and the deployment durations.

VigilNet [8] includes 200 nodes to support military surveillance, covering an area of 100×100 square meters. ExScal [9] attempts to deploy a WSN at a large scale. The system consists of over 1000 sensor nodes and 200 backbone nodes, while it fails to keep in continuous operation for long. Bapat et al. [1] analyze the yield of ExScal. SensorScope [10] is a real-world WSN system on rock glacier, of which the largest deployment consists of nearly 100 sensor nodes. Barrenetxea et al. [11] give practical guidelines for WSN deployments. Liu et al. [2] present measurement results of a large-scale sensor network in the forest. Other deployments include LOFAR-agro [12], PermaDAQ [13], and etc. While most of the works report the overall network delivery performance, they do not examine the correlations among different events, thus fail to investigate the underlying causes of the losses.

There are also some dedicated measurement studies in wireless and sensor networks. Zhao et al. [3] report a measurement study on packet delivery performance using 60 Mica nodes. Srinivasan et al. [4] present measurement of packet delivery performance of the Telos and MicaZ platforms. These measurement studies are important for understanding the impacts of particular factors at different layers. The adopted measurement approaches, however, are targeted at individual aspects and cannot be easily integrated for a synthetical study in an operating WSN since they do not consider the joint impact of many aspects together, such as PHY, LPL MAC, and multi-hop routing, in an operating WSN at a large scale.

For years, network measurement has been a hot topic in the field of Internet, which attracts many research efforts. Wang et al. [14] conduct a measurement study on the impact of routing events on the end-to-end path performance. They show that end-to-end Internet path performance degradation is correlated with routing dynamics and analyze the root cause of the correlation between routing dynamics and such performance degradation. Turner et al. [15] present a methodology for understanding the causes and impact of link failures. They opportunistically mine data sources that are already available in modern network environments and analyze over five years of failure events in a large regional network. The viewpoint of existing Internet measurement studies can be regarded as important references for our work in the WSN context. Nevertheless, understanding the behavior and performance of a WSN is an even more complex and challenging task due to (1) the complex behaviors of the network and its nodes, (2) the lack of common infrastructure for the retrieval of system events, and (3) the insufficient operational efforts for categorizing the losses. Hence, we need a new measurement and analysis approach to understanding the packet delivery performance of a WSN.

III. Data Sources

In order to set the context for our analysis, we briefly describe our system first, and then detail the particular data sources available.

The GreenOrbs network. Our ongoing research project aims at building a long-term and large-scale WSN system in the forest. It employs the TelosB mote [16] with msp430f1611 processor and CC2420 radio. The project was started from April 2009. From August 2010, we rebuilt the software based on TinyOS 2.1.1 [17], with an improved architecture and implementation of the measurement system.

Each node employs the TinyOS LPL MAC, the 4bitle link estimation protocol, the CTP data collection protocol, and the Drip dissemination protocol.

On Dec. 10, 2010, we started a new deployment of the system with nearly 400 nodes in the campus woodlands (with the power level of 31), covering an area of about 60,000m^2. A single TelosB sink node was used for collecting data. The collected data is used to support various forestry applications such as canopy estimate, fire risk prediction, etc. In this paper, we analyze the collected packets of 10 days starting from Dec. 19, 2010. The trace contains 1,137,430 packets in total.

Collected packets. The sink node collects three kinds of packets with CTP collection types C1, C2, and C3, respectively. In the following paragraphs, we introduce the data fields used in our analysis.

The C1 packet contains two kinds of information: (1) sensor data, including temperature, humidity, light, and voltage (2) routing information, including path-ETX [18] from the source node to the sink node, and node IDs along the path (with a maximum number of 10).

The C2 packet contains the routing table with a maximum neighbor number of 10. Each routing table entry contains (1) the neighbor node ID (2) the RSSI value from the neighbor (3) the link-ETX estimate to the neighbor (4) the path-ETX estimate to the sink.

The C3 packet contains various counters: (1) the `CPU` counter records the accumulated task execution time in unit of $\frac{1}{32}$ms (2) the `radio` counter records the accumulated radio-on time in milliseconds (3) the `transmit` counter records the accumulated number of transmitted packets (4) the `receive` counter records the accumulated number of received packets (5) the `drop_no_ack` counter records the accumulated number of packet drops because the retransmission threshold (e.g., 30 in CTP) is exceeded. (6) the `drop_overflow` counter records the accumulated number of packet drops due to queue overflow (7) the `loop` counter records the accumulated number of detected loops.

The abovementioned three kinds of packets also share a common packet header including (1) the `source` field, indicating which node the packet originates from (2) the `seqno` field which increments when CTP sends a packet (3) the `thl` field which indicates the hop count of the arriving packet (4) the `source_time` field which is the time instant when the source node transmits the packet in its local time. (5) the `sink_time` field which is the packet reception time in the local clock of the sink node.

IV. Basic Statistics

With collected packets of 10 days, we are able to extract some basic statistics about the working system. During the measurement period, there are 343 nodes with PDR$\geq$10%. We detect that the sink was down from 14:40 pm Dec. 24, 2010 to 8:20 am Dec. 25, 2010. In our analysis, we exclude the sink-down time and nodes with PDR<10% in order not to bias our analysis.

As each node sends three packets every 10 minutes, we know the number of packets that should be received during the measurement period when the delivery from the source node is fully reliable. Without considering packet losses during the sink-down time, we plot the CDF of the PDR of each node in Figure 2. We can see that there are 48% nodes with PDR>90%, 74% nodes with PDR>80%, and the remaining 26% nodes contribute to 64.3% of the total losses. The system achieves an average PDR of 81.3%. Figure 3 shows the CDF of the radio duty cycle of each node. The average radio duty cycle of the system is 4.9%. Those results indicate that the current ready-to-use TinyOS protocols are not good enough compared to tailored protocols which are reported to achieve 99.9% reliability at permilli in previous works [19], [20].

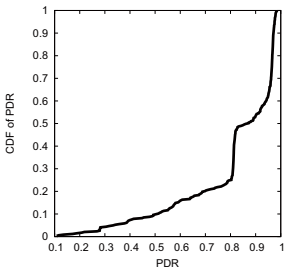


Fig. 2: CDF of PDR

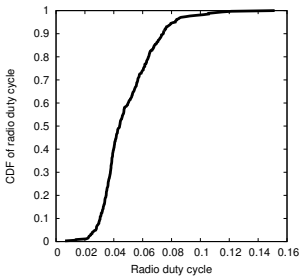


Fig. 3: CDF of duty cycle

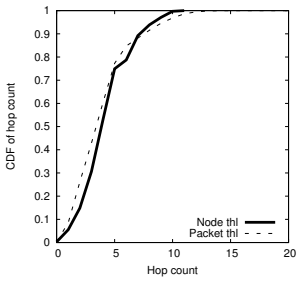


Fig. 4: CDF of hop counts

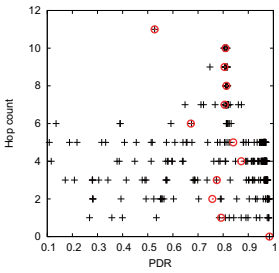


Fig. 5: PDR vs. hop count

Figure 4 shows the CDF of the hop count (to the sink) of received packet and the hop count of each node. We calculate the hop count of each node as the median hop count among all received packets originated from the node. The two curves are close to each other, implying that there is no clear trend that the number of packet losses on long path is higher than that on short path since otherwise the sink should receive more packets with smaller hop counts.

The above implication is further confirmed in Figure 5 which shows the scatter plot of PDR vs. node's median hop count. We also plot the average PDR in each hop count by a red circle. It appears that, for nodes near the sink, the PDRs have a high variance. On the contrary, for nodes far away from the sink, the PDRs are mainly concentrated around 80%. This result implies that bad links do not impact nodes far away from the sink but do impact nodes near the sink.

V. Loss Identification: A First Step

Basically, we identify packet losses by observing gaps in sequence numbers (`seqno`). In our study, we identify loss event of the form `<ID, stime, etime, loss_size>` where `ID` is the source node ID, `stime` and `etime` denote the start time and end time, and `loss_size` denotes the loss size which is the maximum number of consecutive packets lost by the same source node. Two types of packet losses can be identified by observing the `drop_no_ack` and `drop_overflow` counters contained in C3 packets.

We also want to identify other interesting system events, called triggers (e.g., packet corruption, loop, reboot), which can explain the loss events. Each trigger is annotated with the node ID where the trigger is detected, a start time, an end time, and a scope containing a list of impacted nodes. Impacted nodes refer to nodes that are likely to be impacted by the trigger.

Due to space limit, we do not present algorithms for identifying loss events and triggers. Interested readers are referred to [21] for the details.

We have identified a total of 181,862 losses. We have also identified 5,930 no-ack loss events (by examining `drop_no_ack`), totaling 84,030 losses, and 347 overflow loss events (by examining `drop_overflow`), totaling 5,219 packet losses. They contribute to nearly 50% of the identified losses.

Figure 6 shows the loss events during the measurement period. The x-axis denotes the time and y-axis denotes the source node IDs in ascending order. Each loss event is represented by a black line located according to its start time and end time. We also plot the no-ack loss event by a red line and the overflow loss event by a green line. A diamond indicates a reboot event. We can make several observations from this figure.

Vertical banding. We can see two types of vertical banding here. (1) Vertical banding covering all nodes (V1). This happens during 12:54:05, Dec. 25∼16:34:05, Dec. 25, and during 13:03:58, Dec. 28∼13:23:58, Dec. 28. The root cause should be at the sink side. (2) Vertical banding covering a subset of nodes (V2). In this case, a subset of nodes experience packet losses simultaneously. This is mostly caused by routing loops. An evidence is that we also observe overflow losses at a subset of nodes (green lines).

Horizontal banding. We observe that some nodes experience heavy packet losses during the measurement period. Most of them are accompanied with no-ack loss events, suggesting that those nodes may have very poor link qualities to neighboring nodes, causing the retransmission threshold to be exceeded. Interestingly, we observe that most of these nodes experience a recover in the midday. We will further look into such a phenomenon in the following section.

VI. Root Causes: A Closer Look

To investigate the root causes of packet losses, we would like to perform correlation analysis between the loss events and the triggers identified in the previous section. Intuitively, if a loss event is highly correlated with a trigger, it is caused by the trigger with a high probability.

We investigate the following categories of causes:

(A) Sink-side failures which are further classified into
  (A1) sink node failures, and,
  (A2) PC-end failures
(B) Corruptions which are further classified into
  (B1) in-network corruptions, and,
  (B2) corruptions at the sink node
(C) Overflow drops which are further classified into
  (C1) loop-induced overflow drops, and,
  (C2) non-loop overflow drops
(D) No-ack drops which are further classified into
  (D1) environment-induced no-ack drops (env-no-ack drops), and,
  (D2) interference-induced no-ack drops (interference-no-ack drops) which can be induced by node reboot or routing loops. Both node reboot and routing loops will cause a very high beaconing rate in the

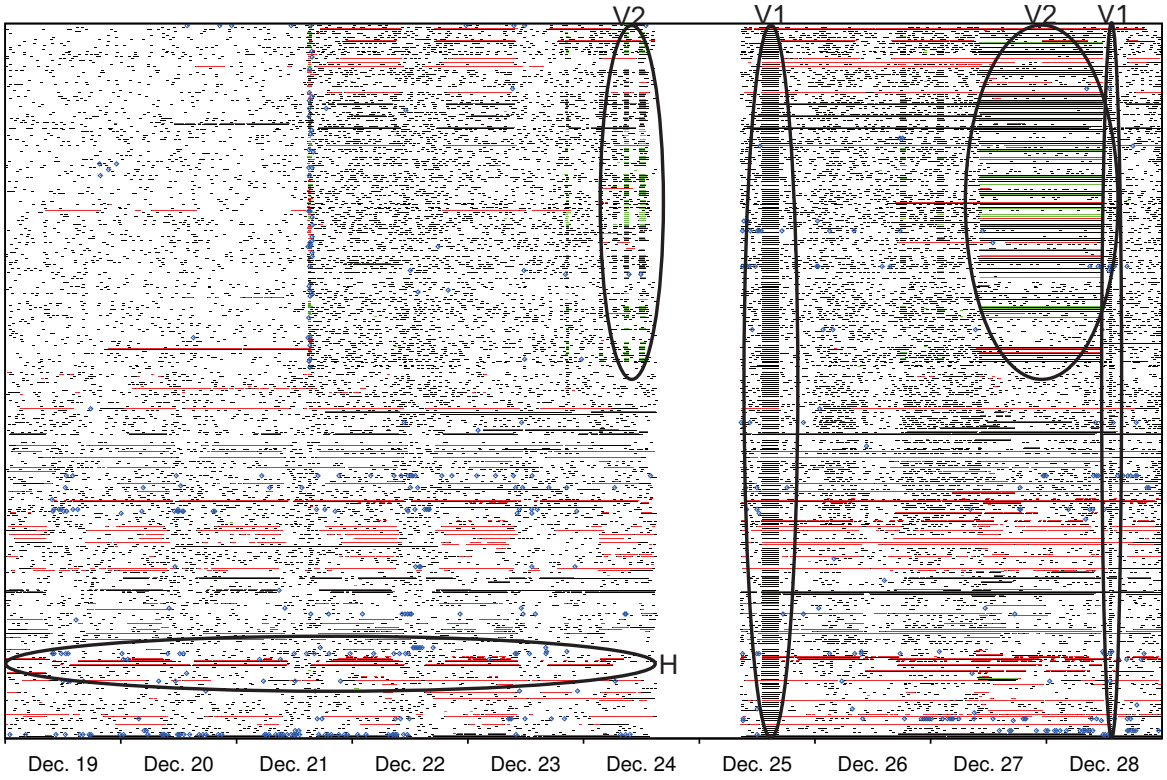


Fig. 6: Packet losses at a glance. The y-axis denotes the node IDs in ascending order. A black line indicates the duration of a loss event. A red line denotes a no-ack loss event and a green line denotes an overflow loss event. A diamond indicates a reboot event. The sink is down during 14:40 pm Dec. 24, 2010 to 8:20 am Dec. 25, 2010.

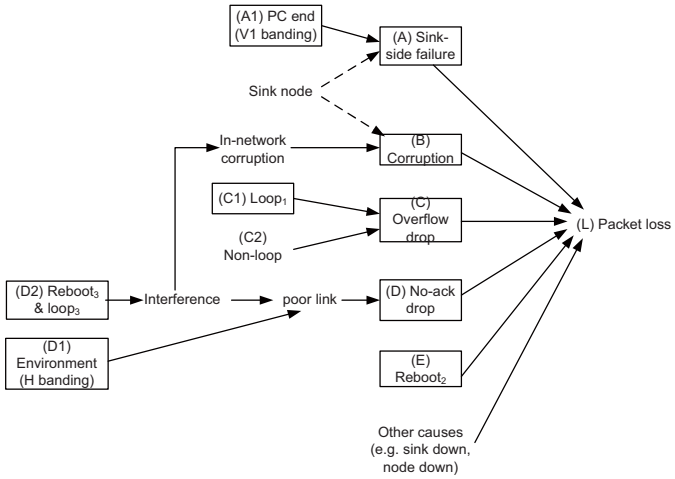


Fig. 7: Causal relationships of system events to loss.

current implementation of CTP/LPL, causing severe interference to neighboring nodes.

(E) Node reboot which can directly cause queued packets of the downstream nodes to be dropped.

There are other causes such as sink down or node down. Figure 7 shows the causal relationships.

We note that a single trigger may have different impacts. For example, a reboot may cause the queued packets to be dropped directly, or, cause interference to all neighboring nodes, resulting in no-ack drops at those nodes. A routing loop may cause overflow drops directly, or, cause interference to all neighboring nodes, resulting in no-ack drops. The multiple impacts of a single trigger and the complex interactions among triggers and the loss events make the delivery performance of a sensor network extremely difficult to reason about.

As each trigger is annotated with a start time and an end time, we use temporal correlation to match it with loss events. To find matches, we widen the start time and end time of a trigger by a time lag to compensate for factors like delayed reporting. The setting of the time lag, however, depends on the scenario under consideration. To minimize the false positives, we also consider the impact scope of a trigger. Each trigger can be denoted as `<ID, stime, etime, scope>` where `ID` denotes the node ID where the trigger is detected, `stime` and `etime` denote the start time and end time of the trigger, and `scope` contains a list of nodes that are likely to be impacted by the trigger.

We consider three kinds of impact scopes in this study:

(1) trigger.scope := trigger.ID.
(2) trigger.scope := downstream nodes of trigger.ID.
(3) trigger.scope := neighboring nodes of trigger.ID.

Detailed process on how to find the scope is described in [21].

We will use a subscript to differentiate triggers with different scopes: trigger_1 has impacts on trigger.ID, trigger_2 has impacts on all downstream nodes of trigger.ID, and trigger_3 has impacts on all neighboring nodes of trigger.ID. In correlating triggers to loss events (or triggers), we ensure that the `ID` in the loss event is contained in the scope of the trigger, i.e., event.ID $\in$ trigger.scope.

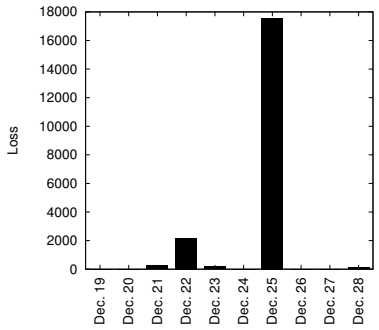


Fig. 8: Sink-side losses for each day.

A. Sink-side failures

The sink node receives packets via the wireless radio, and then forwards the packets via the serial port to the PC where a java tool records the collected packets.

By inspecting the `receive` counter in packets originated from the sink node, we are able to detect that 22,873 packets are dropped at the sink side, i.e., either at the sink node or at the PC. The readers are referred to [21] for the detailed calculation.

What causes sink-side failures? Figure 8 plots the number of packet losses at sink side for each day. We observe that a high number of packet losses happened in Dec. 25. Interestingly, we can see that one vertical banding covering all nodes in Dec. 25 from Figure 6.

Checking A1→A (the labels A and A1 correspond to triggers/events shown in Figure 7). The number of packet losses corresponding to the vertical banding (i.e., V2) is 22,638, which means the vertical banding contributes to 99.6% of the sink-side losses. To further investigate the causes of the vertical banding, we examine the status of the sink node. We find that the sink node does not reboot across the event since the seqno of packets from the sink node continues to increase. After many rounds of detections during testbed experiments, we find that several causes exist: (1) the serial line connecting the sink node and PC is too long and thus is unreliable in delivering packets; (2) the serial port number on the PC unexpectedly changes, causing failure of the java tool; (3) the java tool seems blocked (a restart of the java tool will solve the problem). The above causes are outside the sink node. We collectively call them PC-end failures.

The above result also implies that the amount of packet losses inside the sink node (e.g., due to queue overflow) appears to be small.

B. Corruptions

Corrupted packets are difficult to identify in the first place. We only check a limited packet fields (e.g., the ID field, the routing table entries) to validate the correctness.

We are able to detect a total of 9,511 corrupted packets which correspond to the same number of corruption triggers. This does not necessarily indicate that 9,511 packets are lost because of corruptions. We find that there are 222 corruption triggers that are guaranteed not to cause losses because the following correct packet has exactly the same `source` and `seqno` fields with the previous corrupted packet.

Checking B→L. We try to match the corruption triggers to the loss events to find the actual losses caused by corruption. We do not match for the 127 corrupted triggers with broken `source` fields because those triggers cannot be used for spatial correlation and thus may cause a large false positive. We set the time lag as 10 minutes since a smaller time lag will inevitably cause false negatives because our loss detection latency can reach 10 minutes (i.e., one transmission period). We have detected a total of 9,037 corruption-induced losses. This contributes to 9,037/181,862=5% of the identified loss. In order to get an estimate of the false positive rate, we consider a sample of the 222 corrupted packets that are guaranteed not to cause losses. The matching algorithm finds 19 of those packets correlated with losses, implying an false positive rate of 19/222=8.5%.

Checking D2→B. How do corruptions occur? Does in-network interference cause packet corruptions? To answer these questions, we correlate reboot_3 and loop_3 to the corruption triggers, we find that 3,001 corruptions triggers are correlated with either reboot_3 or loop_3, indicating that 3,001/9,511=31.6% corruptions are highly likely to be caused by in-network interference. This implies that the current packet-level CRC mechanism cannot guarantee the correctness of a receiving packet.

C. Overflow drops

From Figure 6, we can get an initial guess that loops can cause overflow drops as there are many overflow drops (green lines) in occurrence with loops (vertical banding covering a subset of nodes).

Checking C1→C. To take a closer look, we try to correlate loop_1 triggers to overflow loss events. The scope of the trigger only includes loop.ID since in this case the overflowed nodes should also see the loop events if the overflow drops is caused by the loop. As both the triggers and the loss events are identified using C3 packets, we use a small time lag of one second here. We have matched 399 loop_1 triggers to 322 overflow loss events, totaling 5,178 losses.

This result implies several facts. First, overflow drops are mainly caused by loops. Loop-induced overflow drops occupy 5,178/5,219=99.2% (5,219 is the number of identified overflow losses) of the total identified overflow losses. Second, the non-loop overflow drops only occupy 0.8% of the identified overflow losses. We have manually inspected 15 non-loop overflow events and find that nodes experience non-loop overflows have two characteristics: they either experience a sudden increase in the number of received packets, or, they have a high incoming traffic (>300 packets in one period of 10 minutes). Third, loops do not necessarily cause overflow drops: 93% of the loop events do not cause overflow drops. It is, however, possible that loops may cause other kinds of packet losses, e.g., interference-induced losses.

We are interested in the loop events that cause overflow drops. Where and how do they occur? To get a first impression, Figure 9 shows the scatter plot of each node's median hop count vs. detected number of loops that cause overflow drops. The median hop count of each node is calculated by excluding

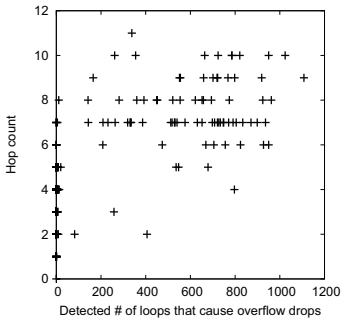


Fig. 9: Node's hop count vs. detected number of loops that cause overflow drops.

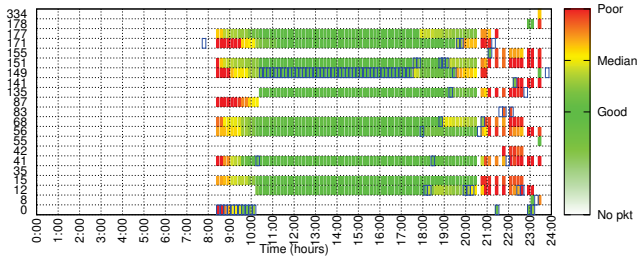


Fig. 10: Node 19's routing table on Dec. 19

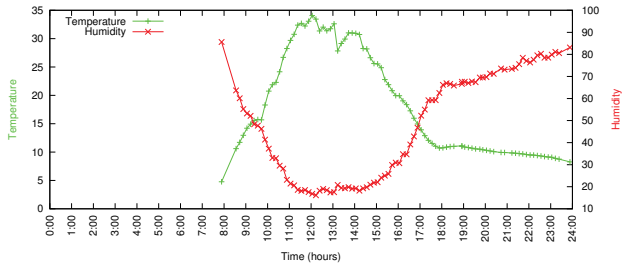


Fig. 11: Node 19's temperature and humidity on Dec. 19

the looping period. We observe that nodes far away from the sink can be easily involved into loops.

D. No-ack drops

No-ack drops constitute the largest portion of packet losses. Conceptually, this type of loss is incurred by poor link qualities to the neighboring nodes, causing the retransmission threshold to be exceeded. There are generally two kinds of factors impacting the link quality used for routing, i.e., physical connectivity and interference. Physical connectivity can be influenced by the environment or the network deployment. Interference is mainly caused by in-network traffic since our system uses channel 15 which does not overlap with WiFi channels.

1) Environment-induced no-ack loss: From Figure 6, we observe that a number of nodes experience serious packet losses (horizontal banding). Interestingly, those nodes experience a recover in the midday. This phenomenon is more obvious in the first six days. To investigate the underlying causes, we inspect a representative node, node 19. Figure 10 shows node 19's routing table on Dec. 19. The x-axis denotes the time and the y-axis denotes the neighboring nodes that

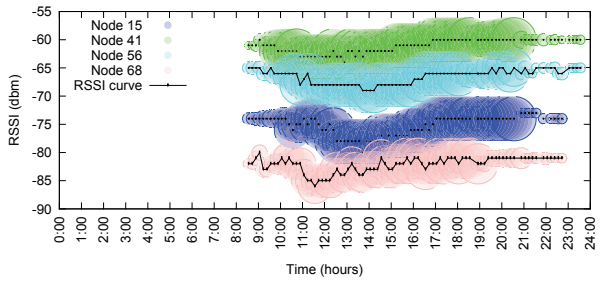


Fig. 12: RSSI and link PRR in node 19's routing table on Dec. 19.

appeared in the routing table at least once during the day. The color represents the link quality to the neighbor. A green color indicates a good link quality, a yellow color indicates a median link quality and the red color indicates a poor link quality. We also show the parent of node 19 by a blue rectangle. The figure does not show information near the start and end times of the day because no C2 packets from node 19 were received. We can see that the link qualities to all nodes experience an abrupt change during 9:00 am–10:00 am (increase) and 20:00 pm–21:00 pm (decrease). It makes us believe that the environment has a large impact on the link quality.

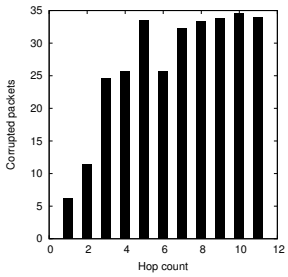
Therefore we plot node 19's temperature and humidity on Dec. 19 in Figure 11. We see that the changes in link quality seem indeed correlated with the environment. This result indicates that the current routing protocols can be greatly improved by using sensor hints and local buffering mechanisms: env-no-ack losses can be largely mitigated by sending packets when the link condition becomes good in the midday.

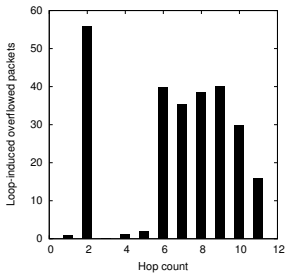
Figure 12 shows the RSSI and link PRR (=1/linkETX) in node 19's routing table on Dec. 19. The radius of the circle indicates the link quality to the neighboring nodes. Interestingly, we see that there is a negative correlation between RSSI and PRR, suggesting that the degradation in link quality is not due to channel fading. Thus, our conjecture is that particles and water pooling on the plastic enclosure are likely to alter the radiation patterns, causing link quality degradations. Such a result confirms similar findings in [22] in which the authors present experimental evidence which demonstrates that changes in link quality are not a result of rain induced fading, but rather due to presence of water.

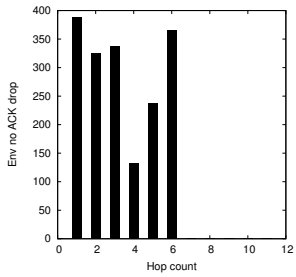
Checking D1→D. We identify env-no-ack losses by identifying nodes that exhibit periodic behaviors in packet losses. There are 68,444 env-no-ack losses, i.e., 37.6% of the total identified losses.

2) Interference-induced no-ack loss: Interference has a large impact on the performance of wireless links. From the CTP/LPL implementation, we know that both the reboot and loop events can cause a high beaconing rate since the Trickle timer will be reset to its minimum interval of 128ms. With LPL, interference will be severe because of long preambles in packet transmissions.

We consider interference caused by reboots and loops in our current study. We do not consider interference caused by data packet transmissions because the exact timing of data packet transmissions is left unknown.







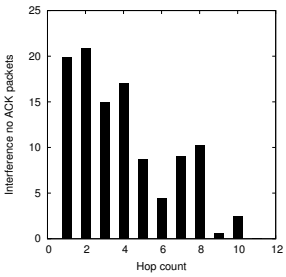


Fig. 13: Packet corruptions **Fig. 14: Loop overflow drops** **Fig. 15: Env-no-ack drops** **Fig. 16: Interf-no-ack drops**

Table 1: Root causes of identified losses

Root cause	**%**
A. sink-side failure	12.5%
A1. PC-end failure	12.45%
A2. sink node loss	0.05%
B. corruption	5%
C. overflow drops	2.87%
C1. loop overflow drops	2.85%
C2. non-loop overflow drops	0.02%
D. no-ack drops	46.2%
D1. env-no-ack drops	37.6%
D2. interference-no-ack drops	2.4%
E. reboot (direct impact on loss)	$\sim$0

Checking D2→D. We try to correlate reboot_3 and loop_3 triggers to no-ack loss events in order to investigate interference-induced no-ack loss. The scope of the reboot and loop triggers are set to be the neighboring nodes which are interfered by the corresponding triggers. We have matched 247 loop events to 536 no-ack loss events, totaling 4,361 losses. We have also found that 10 no-ack losses are matched with reboot events with a time lag of 10 minutes. No loss event is found to be correlated to both reboots and loops. Therefore, the total number of losses correlated with interference is at least 4,371, occupying 2.4% of the identified losses.

E. Summary

We give a summary about the root causes we have found so far. Table 1 gives the root causes and the percentage of identified losses they induce.

There are 33.43% remaining losses we cannot associate with root causes. This is due to several reasons. (1) The identified triggers are not complete. (2) There are false negatives in our matching algorithm. (3) There are other root causes that our measurement data is insufficient to capture.

F. Understanding the Loss

We look at the characteristics of four important losses, i.e., corruption-induced loss, loop overflow drops, env-no-ack drops, and interference-no-ack drops. Here we present the spatial distributions. Interested readers are referred to [21] for the loss size distributions and temporal distributions.

Figures 1(a-d) show the geometric distributions of four categories of losses. Figures 13–16 show the spatial distributions of four categories of loss with respect to the median hop count of each node. Figure 13 shows that while corruption-induced losses increase for the first five hops, it is not apparent for the larger hops. Figure 14 shows that loop overflow drops mainly occur in the nodes with larger hop counts where routing loops can occur more easily. The large value for hop 2 is caused by a single node 576 involved in loops (which drops 1000+ packets because of overflow). Figure 15 shows that env-no-ack drops occur in nodes near the sink. We suspect that it is related to our specific deployment where those nodes are close to a river. Figure 16 shows that interference-no-ack drops mainly occur in nodes near the sink because of high traffic load.

VII. Limitations

Our current data sources are collected in the form of data packets and are relatively easy to retrieve. Nevertheless they cannot capture the complete set of system events in the network. For some wireless behaviors, such as channel utilization, MAC efficiency, additional measurement infrastructures such as passive sniffing or local logging are required. We have attempted to employ sniffers and local logging in our network. But the current storage size can only support a restricted duration for measurement. Also the retrieval of a large number of sniffers or distributed logs is labor-intensive and time-consuming.

Another fact is that the TelosB node used in our current deployment cannot accommodate our application program and the local logging component simultaneously. When the hardware allows, we believe that limited use of local logging and passive sniffing will be useful complement for further investigation into the detailed networking behaviors. We will explore those approaches as future work.

VIII. Implications and Lessons Learned

In this section we give a summary on the observations, implications, and lessoned learned in this study.

Observation 1: The overall delivery performance of our system is 81.3% with a radio duty cycle of 4.9%. **Implication 1**: This indicates that the current ready-to-use TinyOS low power protocol stack is not good enough compared to tailored protocols which are reported to achieve 99.9% reliability at permilli [19], [20].

Observation 2: The number of packet losses on long path is no higher than that on short path. For nodes near the sink, the PDRs have a high variance while for nodes far away from the sink, the PDRs are mainly concentrated around 80%. **Implication 2**: This implies the existence of some bad links since otherwise all PDRs will be high. Those bad links do not impact nodes far away from the sink (distant nodes) but do impact nodes near the sink (nearby nodes). This further implies that the current routing metric can avoid the selection of bad links for distant nodes but cannot optimize the delivery

performance for nearly nodes by enforcing a longer and more stable path.

Observation 3: Sink-side failures are mainly incurred at the PC end instead of the sink node. **Implication 3**: PC-end hardware and software should be closely monitored to minimize packet losses. The use of multiple sinks (including sink node and PC) will be effective in improving the reception reliability.

Observation 4: Packet corruption rates are relatively high (at least 5%). Packets can be corrupted in the network during transmission. **Implication 4**: The current packet-level CRC mechanism is not enough to ensure the correctness of a receiving packet.

Observation 5: Overflow drops are mainly caused by routing loops whereas most loops are transient and show no strong correlation with packet losses. **Implication 5**: Routing loops have different impacts on packet delivery performance. On one hand, it decreases the performance because of queue overflow (and interference). On the other hand, it can salvage transient packet losses. With respect to packet delivery performance, we should eliminate loops that cause overflow drops.

Observation 6: There is a negative correlation between RSSI and PRR. **Implication 6**: Link estimation protocols should use multiple factors to decide the link quality.

Observation 7: The environment has a large impact on packet delivery performance. A number of nodes exhibit highly periodic performance variations because many links severely degrade in the night. However, most of the nodes experience a recover in the midday. **Implication 7**: This result indicates that the current routing protocols can be greatly improved by using sensor hints and local buffering mechanisms: packet losses can be largely mitigated by sending packets when the link condition becomes good in the midday.

Observation 8: We find in our deployment an unnegligible number of node reboots and node failures. It also appears that the poor performance of some wireless links are highly related to our specific deployment where those links are near a river. **Implication 8**: Both sensor node hardware and sensor network deployment have great impacts on the system performance. It is suggested that multiple rounds of indoor testbed experiments and outdoor prototype experiments are conducted before a large-scale and long-term sensor network is deployed.

IX. Conclusion and Future Work

In this paper, we present MAP, a methodology for measuring and analyzing the loss performance of a large operating WSN in the wild. Based on the collected data, we present an approach for uncovering the spatial-temporal distributions of the loss events as well as developing a causal graph with which we perform spatial-temporal correlation analysis for revealing the root causes. We summarize implications and lessons learned and give important guidance to future WSN deployments.

There are multiple dimensions to explore. First, we would like to examine more number of system events, such as link quality changes, routing dynamics. Second, we would like to implement our methodology as a realtime service, augmented with limited use of passive sniffing or local logging for deep examination of wireless behaviors.

Acknowledgment

This work is supported in part by the National Science Foundation of China Grants No. 61202402, 61070155, and 61170213, the Fundamental Research Funds for the Central Universities (2012QNA5007), the Research Fund for the Doctoral Program of Higher Education of China (20120101120179), and NSFC Distinguished Young Scholars Program under Grant No. 61125202.

References

[1] S. Bapat, V. Kulathumani, and A. Arora, "Analyzing the Yield of ExScal, a Large-Scale Wireless Sensor Network Experiment," in *Proc. of IEEE ICNP*, 2005.

[2] Y. Liu, Y. He, M. Li, J. Wang, K. Liu, L. Mo, W. Dong, Z. Yang, M. Xi, and J. Zhao, "Does Wireless Sensor Network Scale? A Measurement Study on GreenOrbs," in *Proc. of IEEE INFOCOM*, 2011.

[3] J. Zhao and R. Govindan, "Understanding Packet Delivery Performance In Dense Wireless Sensor Networks," in *Proc. of ACM SenSys*, 2003.

[4] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis, "Understanding the Causes of Packet Delivery Success and Failure in Dense Wireless Sensor Networks (Technical report SING-06-00)," Stanford University, Tech. Rep., 2006.

[5] R. Szewczyk, J. Polastre, A. Mainwaring, J. Anderson, and D. Culler, "Analysis of a large scale habitat monitoring application," in *Proc. of ACM SenSys*, 2004.

[6] 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 *Proc. of ACM SenSys*, 2005.

[7] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and Yield in a Volcano Monitoring Sensor Networks," in *Proc. of USENIX OSDI*, 2006.

[8] 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. on Sensor Network*, vol. 5, no. 1, pp. 1–39, 2009.

[9] A. Arora, R. Ramnath, E. Ertin, and et al., "ExScal: Elements of an Extreme Scale Wireless Sensor Network," in *Proc. of IEEE RTCSA*, 2005.

[10] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "SensorScope: Out-of-the-Box Environmental Monitoring," in *Proc. of ACM/IEEE IPSN*, 2008.

[11] G. Barrenetxea, F. Ingelrest, G. Schaefer, and M. Vetterli, "The Hitchhiker's Guide to Successful Wireless Sensor Network Deployments," in *Proc. of ACM SenSys*, 2008.

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

[13] J. Beutel, S. Gruber, A. Hasler, R. Lim, A. Meier, C. Plessl, I. Talzi, L. Thiele, C. F. Tschudin, M. Woehrle, and M. Yuecel, "PermaDAQ: A Scientific Instrument for Precision Sensing and Data Recovery in Environmental Extremes," in *Proc. of ACM/IEEE IPSN*, 2009.

[14] F. Wang, Z. M. Mao, J. Wang, L. Gao, and R. Bush, "A Measurement Study on the Impact of Routing Events on End-to-End Internet Path Performance," in *Proc. of ACM SIGCOMM*, 2006.

[15] D. Turner, K. Levchenko, A. C. Snoeren, and S. Savage, "California Fault Lines: Understanding the Causes and Impact of Network Failures," in *Proc. of ACM SIGCOMM*, 2010.

[16] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling Ultra-Low Power Wireless Research," in *Proc. of ACM/IEEE IPSN*, 2005.

[17] TinyOS. [Online]. Available: http://www.tinyos.net

[18] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A High-throughput Path Metric for Multi-hop Wireless Routing," in *Proc. of ACM MobiCom*, 2003.

[19] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection Tree Protocol," in *Proc. of ACM SenSys*, 2009.

[20] R. Musaloiu-E, C.-J. M. Liang, and A. Terzis, "Koala: Ultra-Low Power Data Retrieval in Wireless Sensor Networks," in *Proc. of ACM/IEEE IPSN*, 2008.

[21] W. Dong, Y. Liu, Y. He, and T. Zhu, "Measurement and Analysis on the Packet Delivery Performance in A Large Scale Sensor Network," in *Technical Report, http://eagle.zju.edu.cn/home/eos/dongw/pub/map.pdf*, 2011.

[22] A. Markham, N. Trigoni, and S. Ellwood, "Effect of Rainfall On Link Quality in an Outdoor Forest Deployment," in *Proc. of WinSys*, 2010.