

Delay Measurement in Sensor Networks Using Passive Air Monitoring

Zhengming Bu, Bing Wang and Zhijie Shi
Computer Science & Engineering Department
University of Connecticut, Storrs, CT 06269

UCONN CSE Technical Report BECAT/CSE-TR-07-07

ABSTRACT

Wireless sensor networks have been used in many applications with realtime requirements, e.g., emergency response and medical care. For such applications, characterizing delays inside a network is important to evaluate the performance and guide the deployment of the network. Delay measurement in sensor networks is, however, a challenging problem due to the difficulties to synchronize the clocks at the sensor nodes. In this paper, we develop a methodology that uses passive air monitoring to measure per-hop and end-to-end delays in a sensor network. Our method does not generate any additional traffic in the network and requires no clock synchronization. Using this method, we characterize delays in a sensor-network testbed under various settings, e.g., with different network topologies and amount of medium contention.

1. INTRODUCTION

Wireless sensor networks have been used for many applications that have realtime requirements, e.g., emergency response, plant automation and control, and health care. In such applications, the sensed data have to be transported in *realtime* to end users. Very often, the end users are located far away from the sensor network, e.g., in remote monitoring and surveillance applications. Therefore, the sensed data are first transferred through the sensor network towards a sink, which then forwards data to the end users via an interconnecting network, e.g., the Internet.

In this paper, we consider a sensor network for a realtime application as described above. For such a network, measuring end-to-end delay (i.e., from a sensing node to the end user) and per-hop delay (i.e., delay on each hop of the end-to-end path) is important for network management: end-to-end delays can be used

to evaluate the performance of the network and detect paths with excessive delays; per-hop delays can be used to pinpoint the hops that cause large end-to-end delays. Since the interconnecting network typically has much shorter delays and is more stable than the sensor network, we focus on delay measurement inside the sensor network in this paper.

One way (as often used in a wired network) to obtain end-to-end delay from a source s to a sink t is to first synchronize the clocks at these two nodes. Then node s places a timestamp into the packet when sending it, node t timestamps the packet when receiving it, and the difference of these two timestamps is the transmission delay from s to t . Time synchronization is, however, a challenging task in large-scale sensor networks. Although numerous solutions have been proposed (see survey [13] and the references therein), they typically require (a large number of) message exchanges, which consumes the scarce energy of the sensor nodes. One way to eliminate the need for time synchronization is using half of the RTT between s and t as the one-way transmission delay from s to t . However, this may lead to inaccurate estimates given the asymmetric communication in sensor networks [7, 15, 11].

In this paper, taking advantage of the broadcast medium in wireless sensor networks, we propose using *passive air monitoring* (simply referred to as *air monitoring* henceforth) for delay measurement. The main idea of air monitoring is to place a set of dedicated sensor nodes, called *air monitors*, inside a sensor network. Each air monitor captures and decodes MAC-layer packets in its neighborhood. The captured data can be analyzed locally or transmitted using other communication channels (e.g., Bluetooth in [6]) to a central server. Air monitoring has the advantage that it does not generate any

additional traffic in the network. Furthermore, as we shall see (Section 2), it provides a convenient technique to obtain both end-to-end and per-hop delays.

Our paper makes two main contributions. Firstly, we develop a methodology for per-hop and end-to-end delay measurement using air monitoring. Secondly, we use the methodology to characterize delays in a sensor-network testbed under various settings. We find that, for instance, in a setting with light load and no medium contention, per-hop delays are mostly in [6, 14] ms; when two nodes compete for medium, one node can suffer from significantly larger and more variant delays (e.g., most of its delays are in [6, 70] ms).

As related work, air monitoring has been successfully utilized in wireless LANs (WLANs) for network management and characterization (e.g., [2, 14, 8, 3, 5, 9, 4, 12]). A sensor network differs from a WLAN in that it is a multi-hop ad-hoc network while a WLAN is a single-hop (i.e., from a wireless host to an access point) network with infrastructure support. We are aware of only one study on air monitoring in sensor networks [6], which uses air monitoring for code debugging. To the best of our knowledge, our study is the first on delay measurement and characterization in a sensor network.

The rest of the paper is organized as follows. In Section 2, we describe our measurement methodology. In Section 3, we characterize delays in a sensor-network testbed. Finally, Section 4 concludes the paper and presents future work.

2. MEASUREMENT METHODOLOGY

We first present the problem setting and then describe our methodology to obtain per-hop and end-to-end delays.

2.1 Problem setting

Consider an arbitrary source s inside a sensor network. This source transports the sensed data to sink t . Suppose there are k intermediate nodes along the path from source s to sink t . We denote these nodes as n_1, \dots, n_k . For convenience, we also refer to source s as node n_0 and refer to sink t as node n_{k+1} . We represent the path as $(s = n_0, n_1, \dots, n_k, n_{k+1} = t)$. The hops along the path are indexed from 1 to $(k + 1)$. That is, the i -th hop is the hop (n_{i-1}, n_i) . On hop (n_{i-1}, n_i) , we refer to node n_i as the *parent* of node n_{i-1} and node n_{i-1} as the *child* of node n_i .

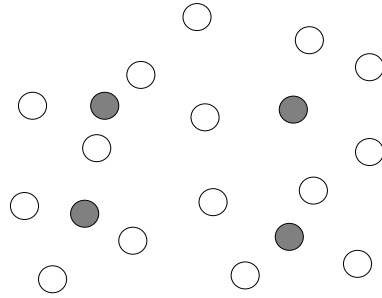


Figure 1: Air monitoring in a sensor network. The shaded nodes are air monitors.

Let D_i denote the delay on the i -th hop, $i = 1, \dots, k + 1$. It contains two components: the delay at node n_{i-1} and the radio propagation delay for a packet to reach node n_i from n_{i-1} . We ignore the latter since the transmission range in a sensor network is tens or hundreds of meters while the radio propagation speed is approximately 3×10^8 meters per second. Therefore, D_i is simply the delay at node n_{i-1} . The delay at source s is from when it sends the packet at the application level to when the sending process is completed. At an intermediate node n_i , the delay is from when the node receives a packet from its child to when the sending process is completed, $i = 1, \dots, k$. Let D denote the end-to-end delay from source s to sink t . It is the delay from when the source sends a packet at the application-level to when the packet reaches the sink. We compute D by adding up the delays over all the hops, that is, $D = \sum_{i=1}^{k+1} D_i$.

We will describe a method that uses air monitoring to obtain per-hop and end-to-end delays. A sensor network with air monitoring is illustrated in Fig. 1, where the shaded nodes are air monitors. Each air monitor captures MAC-level packets in its neighborhood and records the time after capturing a packet. We assume that the air monitors are placed in such a way that, for each hop, there exists at least one air monitor that captures the transmissions from both nodes on the hop.

One advantage of using air monitoring to measure delay is that it eliminates the need to synchronize the clocks at the sensor nodes. This is because an air monitor captures the transmissions from multiple nodes and these transmissions are timestamped according to the same clock, i.e., that in the air monitor. We next de-

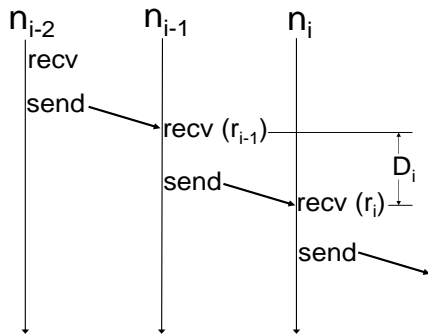


Figure 2: Obtaining delay on the i -th hop, $i \geq 2$ (no packet retransmission).

scribe our methodology to obtain per-hop and end-to-end delay using air monitoring in Sections 2.2 and 2.3.

2.2 Per-hop delay measurement

We describe how to obtain per-hop delay in two cases: (i) no packet retransmission, and (ii) with packet retransmission. In the first case, a node (the source or an intermediate node) only transmits a packet once. In the second case, a node retransmits the packet if not receiving an ACK (all ACKs in this paper refer to MAC-level ACKs) from its parent.

2.2.1 No packet retransmission

We now describe how to obtain per-hop delay without packet retransmission. Consider an arbitrary packet and three adjacent nodes n_{i-2} , n_{i-1} , and n_i , $i \geq 2$, as illustrated in Fig. 2. In the figure, the vertical lines represent time. Let r_i denote the time when node n_i receives the packet from its child, n_{i-1} , $i = 1, \dots, k+1$. Then, since we ignore radio propagation delay, r_i is also the time when node n_{i-1} finishes transmitting the last bit of the packet. Therefore, the delay at node n_{i-1} , which is also the delay on the i -th hop, is $r_i - r_{i-1}$. This delay can be obtained easily through air monitoring. Let T_i be the timestamp of the transmission from node n_{i-1} to n_i that is recorded at the air monitor, $i = 1, \dots, k+1$. Then, we have $D_i = T_i - T_{i-1} = r_i - r_{i-1}$, $i = 2, \dots, k+1$.

The above approach obtains the delay on the i -th hop and requires that $i \geq 2$. For the first hop (s, n_1), we use a different method to obtain the delay. Let source s record a timestamp T_s^b when it is about to send

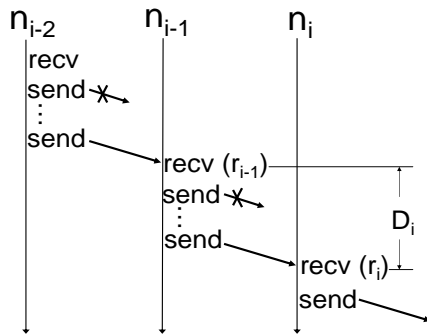


Figure 3: Obtaining delay on the i -th hop, $i \geq 2$ (with packet retransmission).

the packet at the application level, and record another timestamp T_s^e after finishing transmitting the packet. Then the delay at source s , and also the delay on the first hop, is $T_s^e - T_s^b$. Since the first-hop delay is obtained at source s locally, to pass it to the air monitor, source s embeds the first-hop delay of a packet in the payload of the next packet and the air monitor extracts this delay from the payload (we assume a sequence of packets are transmitted from the source).

2.2.2 With packet retransmission

We now describe how to obtain per-hop delay when a packet can be retransmitted. In particular, we again consider an arbitrary packet; each node is allowed to transmit the packet for up to R ($R \geq 2$) times when not hearing an ACK from its parent. In this case, the delay on the first hop can be obtained in the same manner as that in Section 2.2.1. For the delay on a later hop, since a node may transmit the packet for multiple times (see Fig. 3) and hence the air monitor may capture multiple transmissions, the air monitor must know which transmission reaches the parent node successfully. We solve this problem through the DSN (Data Sequence Number) field in the MAC header (specified in the 802.15.4 standard [1]). More specifically, each MAC frame carries a DSN field (of 1 byte) in the header. This field is used to match a data packet and its corresponding ACK (they have the same DSN). For the multiple transmissions of a packet from node n_{i-1} to n_i , we determine the transmission that reaches n_i successfully by identifying the transmission with the DSN field matching that of the ACK from n_i (since a successful transmission will

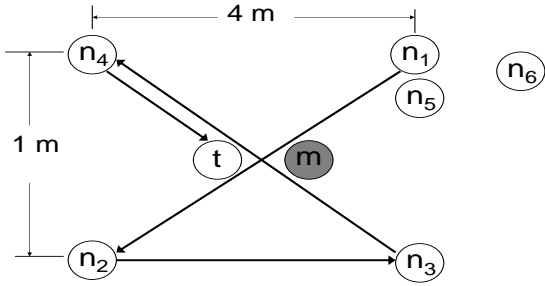


Figure 4: Testbed setting: nodes t and m serve as the sink and air monitor respectively.

trigger an ACK from n_i).

2.3 End-to-end delay measurement

We now describe how to obtain the end-to-end delay from source s to sink t . Let us consider two scenarios. In the first scenario, an air monitor captures the transmissions from all the nodes on the path. In the second scenario, multiple air monitors are used to capture the transmissions from all the nodes on the path.

In the first scenario, we have $D = \sum_{i=1}^{k+1} D_i = D_1 + \sum_{i=2}^{k+1} D_i = D_1 + \sum_{i=2}^{k+1} (T_i - T_{i-1}) = D_1 + T_{k+1} - T_1$, where D_1 is the delay on the first hop (it is embedded in the packets from the source, see Section 2.2.1), T_i is the timestamp of the transmission from node n_{i-1} to n_i at the monitor (when allowing retransmission, it corresponds to the transmission from node n_{i-1} that actually reaches node n_i).

In the second scenario, we segment the path into multiple sub-paths so that a single air monitor captures the transmissions from all nodes on a sub-path. Then, we obtain the end-to-end delay on each sub-path in a similar manner as in the first scenario; and the end-to-end delay of the entire path is the sum of the end-to-end delays on each sub-path.

3. DELAY CHARACTERIZATION

We now apply the delay measurement methodology to measure per-hop and end-to-end delays in a sensor-network testbed. The packets from a source may be lost on the path to the sink. Furthermore, the air monitor does not capture all the transmissions. Therefore, for each setting, we also obtain several statistics on packet loss, packet capture ratio and packet retransmissions.

These statistics, although are not our main focus, help understand the results in each setting. In the following, we first describe the testbed setting and then the various statistics. Last, we present the measurement results.

3.1 Testbed setting

Our testbed consists of eight TelosB motes, as illustrated in Fig. 4. Node t serves as a sink, and node m serves as an air monitor. The air monitor captures and records the time of packet transmissions from all the nodes. The time granularity is 1 ms. Node n_6 is used to synchronize the sources so that they send packets at approximately the same time (as we shall describe soon). The rest of the nodes are sources and/or forwarders. A source sends a packet every two seconds to the sink; each packet has a payload of 28 bytes. A node is allowed to retransmit a packet for up to five times when not receiving an ACK from its parent. The power level at a mote is set to 3, i.e., -25 dBm. The distances between two motes are between 1 to 4 m, as marked in Fig. 4. Therefore, the channel between two motes are in an ideal condition.

We conduct experiments under three settings. The first setting contains a single sender, n_1 , which sends packets via nodes n_2 , n_3 , and n_4 to sink t . This setting represents a desirable operation environment, with light load and no contention in the medium. In the second setting, nodes n_1 and n_5 are sources, both sending packets via nodes n_2 , n_3 , and n_4 to sink t . This represents a setting where contention may occur since the two sources may send to node n_1 simultaneously. We refer to this setting as one with *parallel sources*. In the third setting, nodes n_1 and n_2 are sources and share part of their routes: n_1 sends its packets via nodes n_2 , n_3 , and n_4 to sink t , while n_2 sends its packets via nodes n_3 and n_4 to sink t . This represents a setting in which a sensor node, such as node n_2 in our setting, can be both a forwarder and a source. Contention may also occur in this setting when both sources transmit simultaneously. We refer to this setting as one with *tandem sources*.

In the settings with parallel and tandem sources, when purposely making the sources send at different times, we observe similar delay characteristics as those in the single-source setting. Therefore, in the following, we focus on the scenarios when the sources are synchronized. More specifically, at the beginning of the experiment,

node n_6 sends a signal to both sources. After receiving the signal, these sources set their clocks to zero and start to transmit packets periodically. A scenario with synchronized sources represents an undesirable operation environment. As we shall see, the delay characteristics under desirable and undesirable conditions differ significantly.

We use B-MAC [10], the default MAC protocol in TinyOS. The route from a source to the sink is fixed. In each setting, a source sends 8000 to 12000 packets to the sink. The air monitor runs `apps/TOSBase` that captures MAC-level packets. All the other nodes run `apps/SurgeTelos`. Both programs are modified slightly to fit our purposes. Each packet carries the source and destination address. Furthermore, it carries the address of the node that sends the packet (which may be the source or a forwarder along the path). To differentiate the packets at the application level, each packet carries a sequence number in the payload. Last, for a certain sequence number, to differentiate the multiple transmissions at the application level, we add a field in the payload of a packet: the packet for the i -th transmission has this field as i . This field is used to determine the number of times that a packet is transmitted to reach a parent node successfully.

3.2 Statistics of packet loss and retransmissions

Consider an arbitrary source. Suppose this source transmits N unique packets to the sink. Let \mathcal{L} denote the set of lost packets at the sink. It can be obtained by looking at the sequence numbers of the packets received by the sink. Then the *end-to-end loss rate* is $|\mathcal{L}|/N$. We obtain *per-hop loss rate* as follows. Suppose there are H hops from the source to the sink. Let \mathcal{L}_h represent the set of packets lost on the h -th hop, that is, packets that are transmitted by the child node yet not transmitted by the parent node of the h -th hop (the packets may be lost on the hop or dropped at the parent node), $h = 1, \dots, H$. Then $\mathcal{L}_h \cap \mathcal{L}_{h'} = \emptyset$, $h \neq h'$, $\bigcup_{h=1}^H \mathcal{L}_h \subseteq \mathcal{L}$. We start from the H -th hop (i.e., the hop closest to the sink). Of all the packets in \mathcal{L} , we determine the set of packets lost in the H -th hop, \mathcal{L}_H , and the loss rate on this hop is $|\mathcal{L}_H|/N$. Afterwards, for all the packets in $\mathcal{L} \setminus \mathcal{L}_H$, we determine the set of packets lost on the $(H-1)$ -th hop, \mathcal{L}_{H-1} , and the loss rate on this hop

is $|\mathcal{L}_{H-1}|/N$. We repeat the process till reaching the first hop. For all the packets in $\mathcal{L} \setminus \bigcup_{h=1}^H \mathcal{L}_h \neq \emptyset$, the locations where they are lost are unknown based on the information captured at the air monitor.

Consider an arbitrary node along the path from the source to the sink and the i -th unique packet (i.e., the packet with the sequence number of i) from the node. Let X_i denote the number of times that packet i is transmitted from the node. We infer X_i from the measurement at the air monitor based on the additional field in the packet payload that represents the number of times that a packet has been transmitted (see Section 3.1). (This may provide an underestimate of X_i since the air monitor may not capture all the transmissions). Let $B_i = 1$ denote that the air monitor captures at least one transmission of packet i from the node, and $B_i = 0$ otherwise. Let X denote the total number of packets (including retransmissions) transmitted from the node. Then $X = \sum_{i=1}^N \mathbf{1}(B_i = 1)R_i$, where $\mathbf{1}(\cdot)$ is the indicator function. Let X' be the number of packets captured by the air monitor, $X' \leq X$. Then the *ratio of packets captured at the air monitor* is X'/X . The total number of retransmissions is $\sum_{i=1}^N \mathbf{1}(B_i = 1)(R_i - 1)$. The *ratio of retransmissions* is the total number of retransmissions divided by the total number of transmissions (i.e., X). A retransmission is *unnecessary* if an earlier transmission has reached the parent node successfully. The *ratio of unnecessary retransmissions* is the total number of unnecessary retransmissions over the total number of transmissions.

3.3 Single-source setting

We first report the results under the setting with a single source. Table 1 reports the loss rate on the various hops, the percentage of packets captured at the air monitor, and the percentage of retransmissions and unnecessary retransmissions at the various nodes. A total of 12,839 packets are transmitted from the source to the sink. We observe that the end-to-end and per-hop loss rates are very low.

Fig. 5(a) plots the delay distributions on the four hops from the source to the sink. We observe that these distributions are similar. On the first hop, most delays are in [7, 12] ms, with a mean of 9.5 ms. On the later hops, most delays are in [8, 14] ms, with a mean of 11.0 to 11.3 ms. The variance in the delays might be caused

Table 1: Single source: Packet loss and capture ratio. End-to-end loss rate 0.018.

	loss rate	% of pkts. captured	% of retrans.	% of unnecessary retrans.
from n_1	0.006	82.9%	9.4%	3.0%
from n_2	0.005	95.8%	4.1%	1.5%
from n_3	0.005	96.6%	9.1%	0.3%
from n_4	0.002	99.6%	2.3%	0.0%

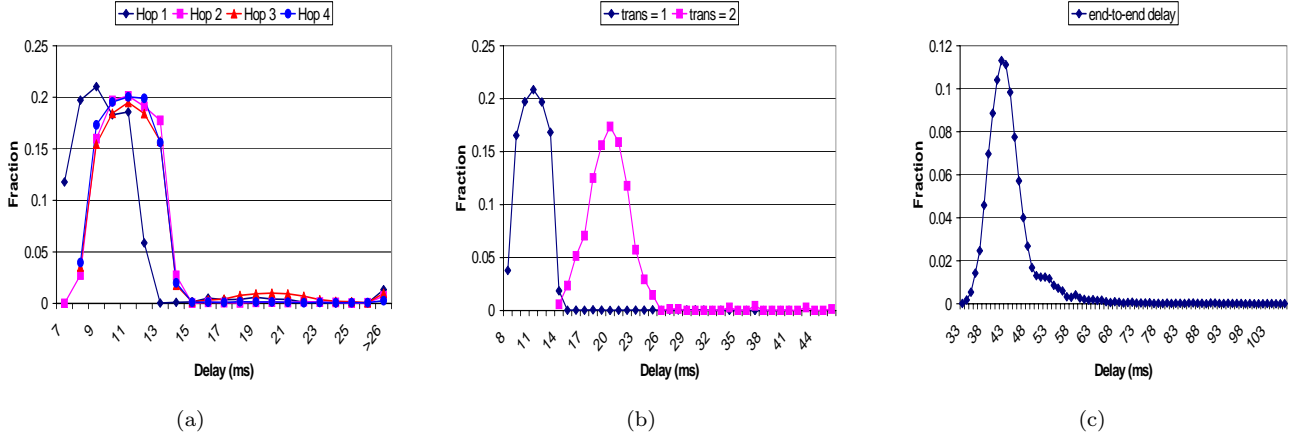


Figure 5: Single source setting: (a) per-hop delay distribution, (b) the conditional delay distribution on the 3rd hop given that the number of transmissions is 1 or 2, (c) end-to-end delay distribution.

by the random delays to access the medium (B-MAC uses carrier sensing). The slightly larger delays on the later hops than those on the first hop might be because the delay on a later hop (defined as the time from when a packet is received to when the sending process of the packet is completed) contains the time for a packet to traverse the protocol stack (from MAC to upper layers), which is not included in the delay on the first hop.

We next investigate the conditional delay distribution on a hop given that a packet is transmitted for i times, $i \geq 1$. On the first and the third hops, around 95% of the packets are transmitted once and 5% of the packets are transmitted twice. On the other hops, around 99% of the packets are transmitted only once. Fig. 5(b) plots the conditional delay distribution on the 3rd hop given that a packet is transmitted once or twice (the results on the 1st hop are similar). We observe that these two distributions are mostly disjoint: when packets are transmitted once, their delays are mostly in [8, 14] ms; when packets are transmitted twice, their delays are mostly in [16, 26] ms. This explains the delays in the range of 16 to 26 ms in Fig. 5(a) for the 1st and 3rd hops.

Last, Fig. 5(c) plots the end-to-end delay distribution. The majority of the delays are in [33, 61] ms, approximately four times of the range (i.e., [8, 14] ms) on a single hop. The mean is 43.9 ms.

3.4 Synchronized parallel sources

We now report the results in the setting with synchronized parallel sources. This setting contains two sources, n_1 and n_5 , and they compete with each other to send packets to node n_2 . Table 2 reports the loss rate on the various hops and the statistics for the various nodes on the path from a source to the sink (the results not in parenthesis are for source n_1 and those in parenthesis are for source n_5). Sources n_1 and n_5 each transmits 6717 packets to the sink. The loss rate of source n_5 is generally much higher than that of source n_1 .

Fig. 6(a) plots the first-hop delay distributions of sources n_1 and n_5 . We observe that these two distributions differ significantly: for source n_1 , the distribution is concentrated in [6, 14] ms, with a mean of 11.7 ms; for source n_5 , the distribution is multi-modal, with a mean of 33.5 ms. The larger delay at node n_5 is due

Table 2: Synchronized parallel sources: Packet loss and capture ratio. End-to-end loss rate 0.09 (0.52).

	loss rate	% of pkts. captured	% of retrans.	% of unnecessary retrans.
from n_1 (n_6)	0.06 (0.37)	87.7% (62.7%)	11.8% (30.9%)	2.5% (1.2%)
from n_2	0.02 (0.09)	85.8% (87.1%)	9.7% (7.9%)	1.8% (1.0%)
from n_3	0.01 (0.05)	86.7% (89.1%)	11.6% (9.9%)	1.4% (0.6%)
from n_4	0.002 (0.002)	86.7% (94.2%)	5.9% (2.5%)	0.1% (0.0%)

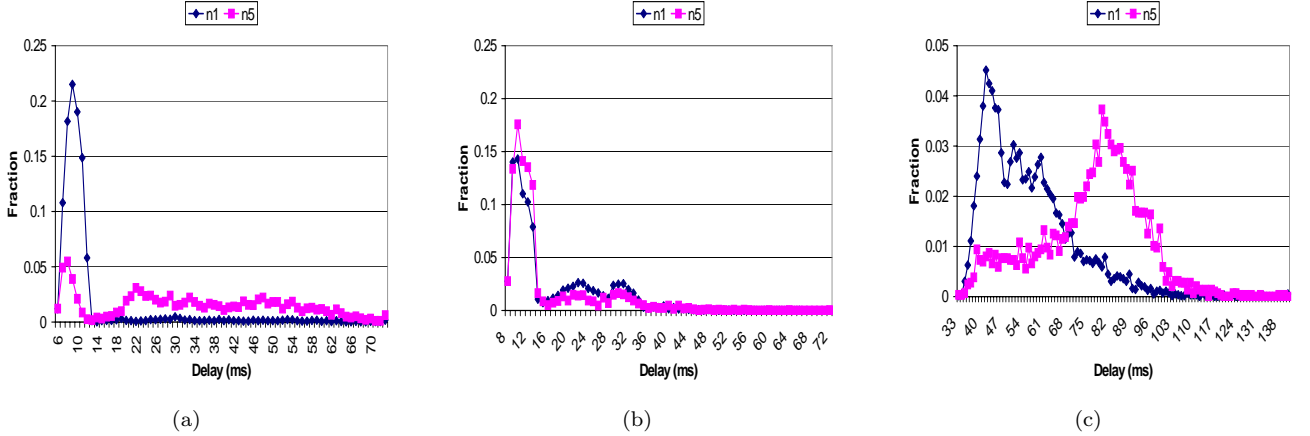


Figure 6: Synchronized parallel sources: (a) first-hop delay distribution, (b) second-hop delay distribution, (c) end-to-end delay distribution.

to more packet retransmissions: for source n_1 , around 95% of the packets are transmitted only once; while for source n_5 , around 60% of the packets are transmitted once, and 30% and 10% of the packets are transmitted twice and three times, respectively. The above indicates that, when two nodes compete for medium, one node can suffer from significantly higher and more variant delays. Looking into the trace at the air monitor, we find that for around 90% of the packets, node n_1 sends the packets earlier than node n_2 . This explains why node n_1 is much less affected by medium contention than node n_5 . We suspect that this is due to physical differences between these two nodes (e.g., the clock at n_1 is slightly faster or n_1 receives the synchronization signal slightly earlier than n_5).

Fig. 6(b) plots the second-hop delay distributions for sources n_1 and n_5 . These two distributions are similar. For both sources, most of the delays are in [8, 36] ms. The average delays for sources n_1 and n_5 are respectively 16.6 and 15.1 ms. Both are higher than that in the absence of medium contention (around 11 ms).

Last, Fig. 6(c) plots the end-to-end delay distributions for sources n_1 and n_5 . Most delays are in the

range of [33, 99] ms. This range is much wider than that without medium contention. The average delays of sources n_1 and n_5 are respectively 55.5 and 76.4 ms. Source n_5 has much larger end-to-end delays than n_1 because of the larger delays on the first hop.

3.5 Synchronized tandem sources

We now report the results in the setting with synchronized tandem sources. This setting contains two sources, n_1 and n_2 . Furthermore, n_2 is also a forwarder for n_1 . Table 3 reports the loss rate on the various hops and the statistics for the various nodes on the path from a source to the sink (the results not in parenthesis are for source n_1 and those in parenthesis are for source n_2). Sources n_1 and n_2 each transmits 7210 packets to the sink. The loss rates of source n_1 on its first two hops are high.

Fig. 7(a) plots the first-hop delay distribution of sources n_1 and n_2 (i.e., on the hops of (n_1, n_2) for n_1 and on the hop of (n_2, n_3) for n_2). For source n_1 , the distribution is multimodal, mostly in [6, 51] ms, with a mean of 20.4 ms; for source n_2 , most of the delays are in [6, 24] ms, with a mean of 14.1 ms. The much larger delay

Table 3: Synchronized tandem sources: Packet loss and capture ratio. End-to-end loss rate 0.54 (0.10).

	loss rate	% of pkts. captured	% of retrans.	% of unnecessary retrans.
from n_1	0.13	68.9%	13.8%	1.2%
from n_2	0.21 (0.05)	83.3% (98.4%)	17.4% (27.1%)	1.1% (2.0%)
from n_3	0.15 (0.002)	94.7% (78.5%)	2.8% (12.3%)	0.0% (0.3%)
from n_4	0.005 (0.04)	99.5% (88.9%)	16.1% (30.7%)	0.1% (0.8%)

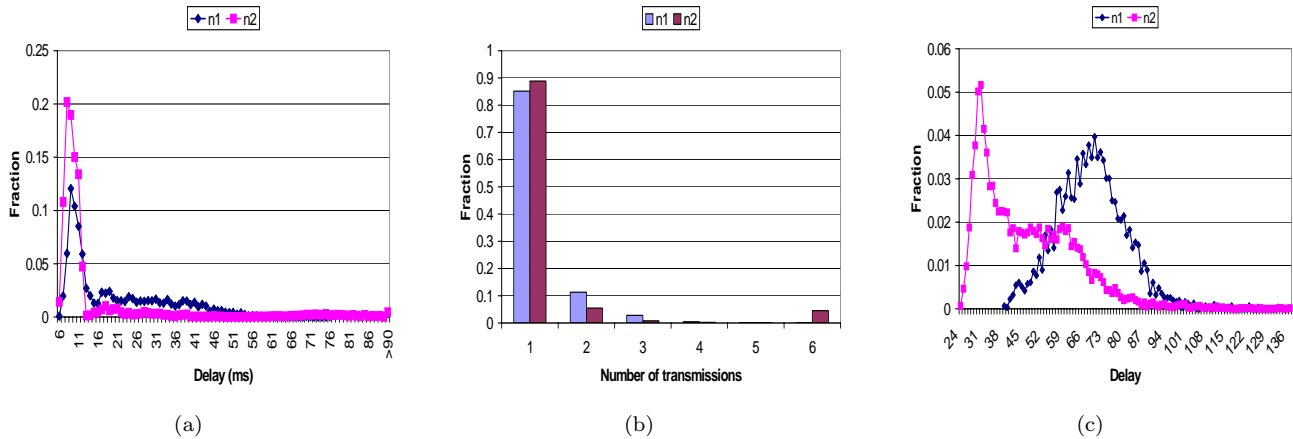


Figure 7: Synchronized tandem sources: (a) first-hop delay distribution (i.e., node n_1 on the hop of (n_1, n_2) , node n_2 on the hop of (n_2, n_3)), (b) distribution of the number of packet transmissions on the first hop, (c) end-to-end delay distribution.

at source n_1 is not caused by more packet retransmissions at n_1 since n_1 only has a slightly larger number of packet retransmissions as shown in Fig. 7(b). Furthermore, it is not due to physical differences between nodes n_1 and n_2 (we switched the nodes that serve as n_1 and n_2 , and observed similar results). We conjecture that the reason might be because n_2 plays a forwarding role for n_1 .

Fig. 7(c) plots the end-to-end delay distribution. For both sources n_1 and n_2 , we observe larger variances than that without medium contention (in Fig. 5). The average delay of source n_1 is 68.8 ms, much higher than that without contention. Source n_2 has only three hops to the sink and hence its average delay is not comparable to that without contention.

4. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a method that uses passive air monitoring to measure per-hop and end-to-end delays in a sensor network. We also used this method to characterize delays in a sensor-network testbed under various settings. We find that, in a setting with

light load and no medium contention, per-hop delays are mostly in [6, 14] ms; when two nodes compete for medium, one node can suffer from significantly larger and more variant delays (e.g., most of its delays are in [6, 70] ms). As future work, we plan to characterize delays in a larger-scale testbed with multiple air monitors.

5. REFERENCES

- [1] IEEE Standard 802.15.4.
<http://www.michaelwhyte.net/zigbee/802154.pdf>.
- [2] A. Adya, V. Bahl, R. Chandra, and L. Qiu. Architecture and techniques for diagnosing faults in IEEE 802.11 infrastructure networks. In *Proc. of ACM MobiCom*, September 2004.
- [3] P. Bahl, R. Chandra, J. Padhye, L. Ravindranath, M. Singh, A. Wolman, and B. Zill. Enhancing the security of corporate Wi-Fi networks using DAIR. In *Proc. of ACM MobiSys*, 2006.
- [4] Y.-C. Cheng, M. Afanasyev, P. Verkaik, P. Benko, J. Chiang, A. C. Snoeren, S. Savage, and G. M. Voelker. Automating cross-layer diagnosis of enterprise wireless networks. In *Proc. of ACM SIGCOMM*, Kyoto, Japan, August 2007.

- [5] Y.-C. Cheng, J. Bellardo, P. Benko, A. C. Snoren, G. M. Voelker, and S. Savage. Jigsaw: Solving the puzzle of enterprise 802.11 analysis. In *Proc. of ACM SIGCOMM*, Pisa, Italy, September 2006.
- [6] F. Dressler, R. Nebel, and A. Awad. Distributed passive monitoring in sensor networks. In *Proc. of IEEE INFOCOM*, 2007. poster.
- [7] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker. Complex behavior at scale: an experimental study of low-power wireless sensor networks. Technical Report UCLA/CSD-TR 02-0013, February 2002.
- [8] A. P. Jardosh, K. N. Ramachandran, K. C. Almeroth, and E. M. Belding-Royer. Understanding congestion in IEEE 802.11b wireless networks. In *Proc. of ACM SIGCOMM Internet Measurement Conference (IMC)*, 2005.
- [9] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Analyzing the MAC-level behavior of wireless networks in the wild. In *Proc. of ACM SIGCOMM*, Pisa, Italy, September 2006.
- [10] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *Proc. of ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2004.
- [11] N. Reijers, G. Halkes, and K. Langendoen. Link layer measurements in sensor networks. In *Proc. of IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, Fort Lauderdale, FL, October 2004.
- [12] A. Sheth, C. Doerr, D. Grunwald, R. Han, and D. C. Sicker. MOJO: A distributed physical layer anomaly detection system for 802.11 WLANs. In *Proc. of ACM MobiSys*, pages 191–204, 2006.
- [13] F. Sivrikaya and B. Yener. Time synchronization in sensor networks: a survey. *IEEE Network*, 18(4), 2004.
- [14] J. Yeo, M. Youssef, and A. Agrawala. A framework for wireless LAN monitoring and its applications. In *Proc. of ACM Workshop on Wireless Security (WiSe)*, 2004.
- [15] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proc. of ACM Conference on Embedded Networked Sensor Systems (SenSys)*,