Network Measurement in Multihop Wireless Networks with Lossy and Correlated Links

Chenhong Cao[1],Wei Gong[2], Wei Dong[1], Jihong Yu[2], Chun Chen[1], Jiangchuan Liu[2]
[1]College of Computer Science, Zhejiang University, China, and,
Alibaba-Zhejiang University Joint Institute of Frontier Technologies
[2]School of Computer Science, Simon Fraser University, Canada
{chcao,dongw,chenc}@zju.edu.cn, {gongweig,jihongy,jcliu}@sfu.ca

***Abstract*—Multihop wireless networking is a key enabling technology for interconnecting a vast number of IoT devices. Measurement is fundamental to various network operations including management, diagnostics, and optimization. Out-of-band measurement approaches use external sniffers to monitor the network traffic passively, and they provide detailed information about the network. However, existing approaches do not carefully consider lossy and correlated links which are common in low-power wireless networks, resulting in unsatisfactory packet capture ratio and low measurement quality. In this paper, we present NetVision, a practical out-of-band measurement system with special consideration for sniffer deployment. By explicitly considering link quality and link correlation, we are able to achieve a high measurement quality while minimizing the deployment cost. We formulate the sniffer deployment problem as an optimization problem and propose efficient algorithms for solving this problem. We further design a set of instructions and APIs to simplify a variety of common measurement tasks. We implement NetVision on the TinyOS/TelosB platform and evaluate its performance extensively both in simulation and an indoor testbed with 80 TelosB nodes. Results show that NetVision is accurate, generic, and robust. Three typical case studies demonstrate that NetVision can facilitate various measurement and debugging tasks.**

I. INTRODUCTION

One of the most important features of IoT (Internet of Things) is to interconnect a vast number of tiny objects with embedded devices. Low power and short-range wireless networking (e.g., ZigBee, Bluetooth) is a suitable technology which is both low cost and energy efficient [1]. With a single-hop network topology, the short transmission range prevents the use cases of IoT applications from running over a large number of devices. Therefore, considerable efforts have been spent in providing multihop capabilities for these short-range wireless technologies. Examples include:

- IETF RoLL has proposed RPL [2] which is a multihop routing protocol built on IPv6 for low-power and lossy networks. RPL has been implemented for wireless sensor networks with 802.15.4 radios. RPL can also be ported to other short-range wireless technologies, e.g., BLE [3].
- WirelessHART is another popular multihop routing protocol which provides real-time and high-reliability guarantees. It is a kind of wireless mesh network communication standard and is widely used in industrial applications [4].
- Recently, Bluetooth SIG has released Bluetooth 5.0 with multihop mesh networking capabilities [5].

In such multihop wireless networks, measurement has become fundamental to network operations including management, security, diagnostics, optimization or simply enhancing our collective understanding of the network as a complex system. Over the years, many network measurement approaches have been proposed [6], [7], [8], [9]. They can be broadly classified into two categories: in-band and out-of-band.

In-band measurement approaches [6], [7], [10] collect the measurement data on the same communication channel for data packets. Although they do not require external infrastructures, they may introduce new network failures due to additional measurement traffic, which is also known as Heisenbugs [11]. More importantly, in-band approaches are vulnerable to possible packet losses since they are limited to observe network states only at the base station.

In contrast, *out-of-band approaches* [8], [9], [12] use external sniffers to monitor the network traffic passively. Compared with in-band approaches, they can provide more detailed information about the network. Previous works have demonstrated that out-of-band approaches are indeed useful for analyzing sensor network behaviors [12], and 802.11b enterprise network behaviors [8], [9].

A key problem for out-of-band measurement is where to deploy the sniffers. For this problem, it is important to introduce the concept of *packet capture ratio* for each node, i.e., the ratio of packets captured (received) by the sniffers among all packets transmitted. Packet capture ratio directly relates to measurement quality: as more packets are captured, more accurate measurement results can be achieved. For example, for path trajectory reconstruction, we need to infer path trajectories of interest from the observations at the sniffers [8], [9], [12]. It is well-known that packet routing behaviors change often due to the dynamic forwarding policies or topology changes in multihop wireless networks [13]. Therefore, it is difficult to infer the packet trajectories accurately with a low packet capture ratio, and more captured packets can offer better inference accuracy. Note that, there is always a tradeoff between the number of deployed sniffers (reflects the deployment cost) and the packet capture ratio (reflects the measurement quality).

We find that existing approaches are unsuitable for multihop wireless networks with lossy and correlated links. Most existing approaches [14], [15] are based on the too ideal unit

978-1-5386-4128-6/18/$31.00 ©2018 IEEE

disk graph (UDG) model, i.e., a sniffer can capture most of the packets sent by nodes located in the circular communication region centered at the sniffer. They cannot accurately quantify the amount of packets captured by the sniffers since the communication range is irregular and the packet capture is probabilistic in practice. In addition, existing approaches do not consider the existence of link correlation which has been shown by many recent studies [16], [17], especially for networks operating in the 2.4GHz ISM bands (e.g., ZigBee and Bluetooth) which could be severely interfered by other wireless technologies (e.g., WiFi). Consider a network node A and two sniffers S1 and S2. The packet capture ratios (i.e., packets captured by S1 and S2) of node A could be quite different under different link correlations.

In order to address these limitations, we present NetVision, a practical network measurement system with special consideration for sniffer deployment. The key insight of NetVision's sniffer deployment algorithm is to explicitly consider the link quality and link correlation between network nodes and sniffer nodes, with the assumption that sniffer nodes can only be deployed on the positions of network nodes. As such, NetVision can accurately quantify the packet capture ratio of each node for a given sniffer deployment. We then formulate the sniffer deployment problem as an optimization problem to minimize the number of sniffers while the packet capture ratio for each node is satisfied. We prove that it is NP-hard, even under the link independent model.We propose efficient heuristic algorithms for this problem.

Based on the complete measurement trace merged from multiple sniffers, NetVision provides a set of instructions and APIs to simplify a variety of measurement tasks, e.g., hotspots analysis, link loss ratio measurement and packet loss localization. Although we currently implement NetVision in the context of wireless sensor network, we envision that its principles can be applied elsewhere.

The main contributions of this paper can be summarized as follows:

- To the best of our knowledge, we are the first to formally consider the sniffer deployment problem in multihop wireless networks with lossy and correlated links.
- We quantify the measurement quality using a metric called packet capture ratio. Based on this metric, we formulate the sniffer deployment problem as an optimization problem and propose efficient heuristic algorithms to address it.
- We design and implement NetVision, a practical network measurement system in TinyOS/TelosB platform. NetVision exposes a set of APIs to facilitate a variety of measurement tasks. Applying NetVision in three case studies demonstrates its benefits over existing in-band measurement approaches.

II. RELATED WORK

Network measurement in multihop wireless networks continues to grow in importance. Researchers and operators have made tremendous efforts on the network measurement to understand multihop wireless network behaviors and operations.

'Out-of-band' measurement: Jigsaw[8] and Wit[9] use external sniffers to monitor the network traffic passively. There, the authors target at monitoring 802.11 networks which mainly adopt single-hop communication and do not take sniffer deployment into consideration. In NetVision, we focus on optimizing the sniffer deployment in multihop wireless networks which exhibit more complex dynamics due to multihop routing and coordinated behaviors across nodes. LiveNet [12] is designed for the wireless sensor network (WSN), a typical case of multihop wireless networks. We also implement and evaluate NetVision in the WSNs. Our work differs in that we solve the key problem about sniffer deployment in practice and facilitate easy customization on a variety of measurement applications. Our techniques can also be applied to many other multihop wireless networks.

There are also related works on out-of-band measurement focusing on sniffer-channel assignment problems [18]. These works are complementary to our work in that they make a hypothesis of a given set of sniffers. In our work, we focus on the sniffer deployment problem minimizing the deployment cost. Their sniffer-channel assignment schemes could be employed in the sniffers of our system.

Sniffer deployment: Several other techniques attempt to address the sniffer deployment problem. SMSN [14] selects sensor nodes from the network and activates sleeping nodes as monitors. It aims at finding an optimal set of monitors to monitor every communication link. A link is monitored if both end nodes of the link adjacent to a monitor. Note that, the monitor cannot be the end nodes of the link. With this requirement, if a node has only one neighbor, then the link between the node and its neighbor cannot be monitored. Another work DMWSN [15] deploys external sniffers at arbitrary locations to ensure that every communication link is monitored. However, monitoring all communication links cannot provide a guarantee on the *packet capture ratio*. Moreover, these works assume the network complies with the UDG model, which is not practical for in-situ networks. Our work has two key differences. First, our system is based on the optimal sniffer deployment problem, which aims at minimizing the number of sniffers and guaranteeing the packet capture ratio of each node. Second, we deploy sniffers at node positions in the network without assuming the UDG model.

'In-band' measurement: Many 'in-band' techniques have been proposed for measurement in multihop wireless networks. [10] implants agents into sensor nodes to periodically send network statuses to the sink. Specifically, it collects the routing table as well as the flow information. Others [6], [7] directly embed the measurement data into the packet header space of sensing data packets and piggyback them to the sink. PAD [6] tracks the packet path by adding a two bytes field to each packet which is used to store one forwarder along the path. Domo [7] attaches a small overhead to each packet to record the accumulated delays at each hop. In contrast, NetVision is an out-of-band approach that does not require any modification of the original code and does not change the current traffic pattern.

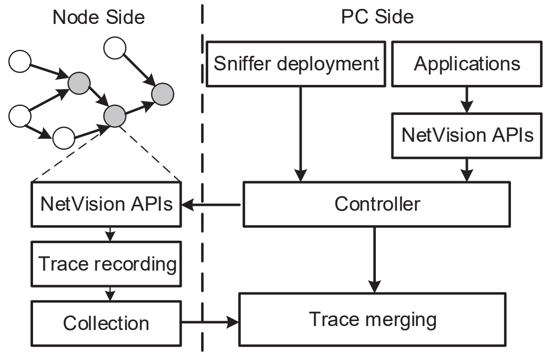


Fig. 1: The overview of NetVision.

III. SYSTEM ARCHITECTURE

NetVision aims to provide a practical and generic network measurement system. Fig. 1 shows its overall architecture. We first introduce NetVision's workflow and then identify the major design goals.

NetVision workflow. At the PC side, NetVision carefully selects positions to place sniffers so that the deployment cost is minimized and the monitoring requirements are satisfied. Once sniffers are deployed to the node side, NetVision starts monitoring the network traffics at each sniffer. The trace recording component records the packet transmission information and the packet arrival time. NetVision APIs on the sniffer allow the user to specify the type of packets they want to monitor. The sniffer traces are then collected to the PC side. At the PC side, the trace merging component merges multiple sniffer traces with the correct transmission information and synchronized timestamps. Based on the merged trace, a set of APIs are exposed by the controller. These APIs can support a variety of measurement applications.

NetVision design goals. We identify the major design goals as follows.

- Deployment efficiency. Our first goal is to achieve deployment efficiency. More specifically, while the packet capture ratio for each node satisfying, NetVision should minimize the number of sniffers. We formulate the sniffer deployment problem into two subproblems, taking link quality and link correlation into consideration. The formulations and solution are described in Section IV.
- Ease of measurement. Our second goal is to allow easy network measurement. NetVision makes measurement programming easier by carefully designing a set of simple instructions and APIs to simplify the development and implementation of many measurement applications. We describe NetVision instructions and APIs in Section VI and provide several examples of using NetVision interfaces for network measurement.

IV. SNIFFER DEPLOYMENT OF NETVISION

A. Assumptions and Definitions

We assume that sniffers are deployed at the positions of network nodes. With the help of this assumption, the quality of links that connect nodes and sniffers can be approximated as that of links between the monitored nodes and nodes at the sniffer position. This assumption also allows us to use the network nodes as sniffers directly. In addition, deploying sniffers at node positions is more practical since not all positions are geographically reachable. For the convenience, we use "sniffer node" to refer "the node whose position is selected as a sniffer position".

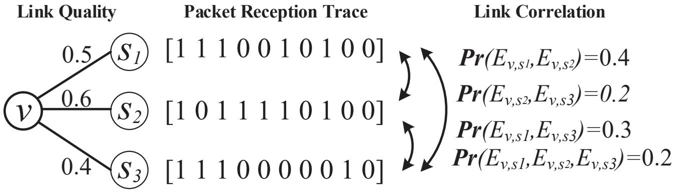


Fig. 2: Illustrative example.

We consider the network as an undirected graph $G(V,E)$, where V and E are the sets of nodes and links. Let S denote the set of sniffers, and $s \in S$ denote a sniffer. Let $C(v)$ denote the packet capture ratio of node $v \in V$. We introduce the concept of *minimum-κ-covered* to guarantee the packet capture ratio of each node.

Definition 1 (Packet Capture Ratio). *The **packet capture ratio** (PCR) of a node is the percentage of its packets observed by sniffers in S. These packets are either originated or forwarded by the node.*

Definition 2 (Minimum-κ-covered). *A node is **minimum-κ-covered** if the packet capture ratio of the node is no less than a specified threshold κ. The network is minimum-κ-covered if all nodes are minimum-κ-covered.*

B. Problems Statement

In the sniffer deployment problem, we aim at deploying a set of sniffers that: (1) the number of sniffers is minimized, (2) the network is minimum-κ-covered, i.e., the packet capture ratio (PCR) of each node is no less than κ. This problem can be quite different under different link models. In many previous studies [19], researchers assume that the wireless links are independent. Under this model, node PCR only depends on the link quality. However, recent studies [16], [17] show that link correlations exist among multiple links which can result in different node PCRs under different link correlations.

Illustrative Example. Fig. 2 shows an illustrative example of sniffer deployment under different link models. In the figure, there is one node and three candidate sniffer nodes. The goal is to select a minimum number of sniffer nodes from s_1, s_2, s_3 to make v minimum-κ-covered. We set the PCR threshold κ to be 0.8. Suppose E_{v,s_i} is the event that a packet is successfully observed by s_i and $\overline{E}_{v,s_i}$ represents that s_i misses the packet. The packet delivery ratio p_{v,s_i} of each link (v, s_i) is given in the figure.

- Under link independence model. According to the link quality, none of the three sniffers can make v minimum-κ-covered. Let us consider sniffer sets of size 2. The PCR of v

with sniffer set $\{s_i, s_j\}$ is the probability that at least one of them observes the packet, i.e., $1-(1-p_{v,s_i})*(1-p_{v,s_j})$. With sniffer set $\{s_1, s_2\}$, the PCR of v is $1-(1-0.5)*(1-0.6)=0.8$. Similarly, the PCR of v with $\{s_2, s_3\}$ and $\{s_1, s_3\}$ are 0.76 and 0.7. As a result, s_1 and s_2 are selected to be sniffer nodes.

- Under link correlation model. For a single sniffer, the node PCR is the same as that under link independence model. For sniffer sets of size 2, the node PCR of v with sniffer set $\{s_i, s_j\}$ is $1-Pr(\overline{E}_{v,s_i}, \overline{E}_{v,s_j})$. With sniffer set $\{s_1, s_2\}$, $Pr(\overline{E}_{v,s_1}, \overline{E}_{v,s_2})$ equals 0.3 from the packet reception trace. Therefore, the PCR of v is 0.7. Similarly, the PCR of v with $\{s_2, s_3\}$ and $\{s_1, s_3\}$ are 0.8 and 0.6. As a result, selecting $\{s_2, s_3\}$ is the best choice.

This example demonstrates the difference of sniffer deployment under different link models. We can see that the link independence assumption sometimes overestimates the node PCR when links are correlated. In this example, the real node PCR is 0.7 with $\{s_1, s_2\}$.

Accordingly, we consider two link models and aim to address the following problems:

- *Link-quality-aware sniffer deployment*. For a given network, assume that the links are independent and the quality of each link is known, we consider the problem of deploying a minimum number of sniffers to ensure the network is minimum-κ-covered.
- *Link-correlation-aware sniffer deployment*. Assume that the links are correlated and the link correlations can be obtained, we consider the same goal as to deploy a minimum number of sniffers to ensure the network is minimum-κ-covered.

C. Problem Formulation

1) Link-quality-aware sniffer deployment: We first investigate the key metric, i.e., node PCR, under the link independence model. In this model, the packet missed by a sniffer has no relationship with the packet missed by other sniffers. A packet is unobserved if and only if all sniffers miss the packet. Let S_v denote the set of sniffers deployed in node v's neighborhood. Let p_{v,s_i} denote the packet delivery ratio from node v to sniffer $s_i \in S_v$. Then, the PCR $C(v)$ of node v can be calculated as:

$$C(v) = 1 - \prod_{s_i \in S_v} (1 - p_{v,s_i}). \tag{1}$$

For example in Fig.2, if sniffers are deployed at s_1 and s_2, the PCR of v is $1-(1-p_{v,s_1})*(1-p_{v,s_2}) = 1-0.5*0.4 = 0.8$.

In the sniffer deployment problem, given a node v, we expect that the packet capture ratio of v is no less than κ, i.e., $C(v) \geq \kappa$. Given the input routing topology $G=(V,E)$ with n nodes, the PCR threshold κ, and the packet delivery ratio $q_{u,v}$ of each link $(u,v) \in E$, let $x_i, i \in \{1,2,...,n\}$, be the binary indicator of whether the i^{th} node is selected as a sniffer node,

$$x_i = \begin{cases} 1 & \text{if } v_i \text{ is selected,} \\ 0 & \text{else.} \end{cases} \tag{2}$$

We can formulate the link-quality-aware sniffer deployment problem as follows.

Problem 1 (Link-quality-aware Sniffer Deployment)**.**

$$\min \sum_{1 \leq i \leq n} x_i$$

$$s.t. \begin{cases} 1 - \prod_{i=0}^{n} (1 - p_{v,i} * x_i) \geq \kappa, & v \in V \\ x_i = 0 \quad or \quad x_i = 1, & i = 1, 2, ..., n \end{cases} \tag{3}$$

Our goal is to find the values of x_i to minimize the number of sniffers. The constraint is that for any node v, it is minimum κ covered.

Theorem 1. *The link-quality-aware sniffer deployment problem is NP-hard.*

Proof. We prove this by a reduction from the Minimum Dominating Set Problem. The proof can be found in the longer version of this paper [20]. ☐

2) Link-correlation-aware sniffer deployment: The link-correlation-aware sniffer deployment differs from link-quality-aware sniffer deployment in the computation of node PCR. We adopt the correlation model proposed in TWC15[21]. The probability that at least one sniffer in S_v receives the packet from v is calculated as:

$$C(v) = \sum_{k=1}^{|S_v|} (-1)^{k-1} Pr(s^k), \tag{4}$$

where $s^k \subset S_v$ is any sniffer set with size k (among $|S_v|$ sniffers) and $Pr(s^k)$ is the probability that any k of the $|S_v|$ sniffers successfully receive a packet. $Pr(s^k)$ is calculated as follows:

$$Pr(s^k) = \sum_{s^k \subset S_v} Pr(E_{v,s_1^k}, ..., E_{v,s_k^k}). \tag{5}$$

For example in Fig.2, assume that all s_1, s_2 and s_3 are selected as sniffer nodes, then the PCR of v is:

$$\begin{aligned} C(v) = p_{v,s_1} + p_{v,s_2} + p_{v,s_3} - Pr(E_{v,s_1}, E_{v,s_2}) \\ - Pr(E_{v,s_1}, E_{v,s_3}) - Pr(E_{v,s_2}, E_{v,s_3}) \\ + Pr(E_{v,s_1}, E_{v,s_2}, E_{v,s_3}), \end{aligned} \tag{6}$$

E_{v,s_i} is the event that a packet is successfully received by sniffer s_i. $Pr(E_{v,s_1}, E_{v,s_2})$ is the probability that both s_1 and s_2 receive the packet, which can be get from the packet reception traces as shown in Fig. 2. The packet reception trace is a sequence of binary bits representing the reception status of a fixed number of most recent packets. The "0" indicates a packet loss and "1" indicates a packet reception. In this example, $Pr(E_{v,s_1}, E_{v,s_2}) = (1\&1 + 1\&0 + ... + 0\&0)/10 = 0.4$. With three sniffers s_1, s_2 and s_3, $C(v) = 0.5 + 0.6 + 0.4 - 0.4 - 0.3 - 0.2 + 0.2 = 0.8$.

Given the input routing topology $G=(V,E)$ with n nodes, the PCR threshold κ, and the packet delivery ratio $q_{u,v}$ of each link $(u,v) \in E$, the link correlation information among links, the link-correlation-aware sniffer deployment problem is presented as follows.

Problem 2 (Link-correlation-aware Sniffer Deployment)**.**

$$\min \sum_{1\leq i\leq n} x_i$$

$$s.t. \begin{cases} \sum_{k=1}^{|S_v|} (-1)^{k-1} Pr(s^k) \geq \kappa \\ S_v = \{v_i | x_i = 1, (v, v_i) \in E\}, \quad i = 1,2,...,n \\ x_i = 0 \quad or \quad x_i = 1, \quad i = 1,2,...,n \end{cases} \tag{7}$$

D. Algorithm Design

We propose a simple greedy algorithm that can achieve a $(\ln\delta + 2)$-approximation ratio. The core idea is to greedily choose a sniffer node that covers as many uncovered neighbors as possible. Note that, we distinguish the concepts of "monitor" and "cover" in this paper: a node is *monitored* if any of the sniffers is deployed in its neighbor; a node is *covered* if it is minimum κ covered. Two or more sniffers can jointly cover one node.

Algorithm 1 presents the pseudo-code of our sniffer deployment algorithm. We use S to represent the set of selected sniffer nodes. We start with $S = \emptyset$ and add nodes to S until the network is minimum-κ-covered. *CoveredSet* is used to store the successfully covered nodes. *CandidateSet* is the set of candidate sniffer nodes and is initialized as V. We greedily choose a sniffer from *CandidateSet* that covers as many nodes as possible in line 6-13. For each node in the *CandidateSet*, we calculate the newly covered nodes in line 8 by adding it to the sniffer set. The function G.getCoverSet($newS$, κ) calculates each node's PCR according to Eq. 1 and returns a set of sensor nodes that are covered by *newS*. The algorithm stops when *CoveredSet* contains all the nodes in the network.

Theorem 2. *The Link-quality-aware Sniffer Placemen Algorithm computes a* $(\ln\delta + 2)$*-approximation, that is, for the computed sniffer set S and an optimal sniffer set* S^**, we have* $\frac{|S|}{|S^*|} \leq \ln\delta + 2$*, where* δ *is the maximal degree of G.*

Proof. The proof can be found in the longer version of this paper [20]. □

Algorithm 1 is also appliable for the Link-correlation-aware Sniffer Deployment. The difference lies in the calculation of the *CoveredSet* which relies on the computation of node PCR. Given the target network G, the sniffer set S, the PCR threshold κ and the link correlation information as inputs, the *CoveredSet* is calculated based on Eq. 4 and 5.

V. Practical Issues

In the previous section, we have introduced the sniffer deployment strategies of NetVision. After that, NetVision starts monitoring the network traffic at each sniffer. The collected traces from multiple sniffers are then merged at the PC side and form the basis of additional measurement and diagnosis. However, there exist several practical issues to implement NetVision in real networks.

Knowledge of link quality and link correlation. Initially, NetVision needs the help of in-band techniques to obtain the link qualities and link correlations. It exploits the broadcast beacons of existing protocols, e.g., CTP [23]. During broadcasting, each node maintains a beacon reception bitmap (e.g., [1010]) recording the reception status of a fixed number (e.g., 4) of most recent beacons. With the beacon reception trace, link quality is simply computed by the number of 1s in the bitmap divided by the bitmap length. The link correlation between multiple links can be computed as described in section IV-C2. The memory overhead of the beacon reception trace relates to the network topology. In our simulation of a 144-node network, with node degree from 10 to 60, we measure the beacon reception trace for 3 hours at each node. The memory overhead is 3.2KB on average which is acceptable for a TelosB node with 1M flash memory.

Algorithm 1 Link-quality-aware Sniffer Deployment Algorithm

```
Input: The targeted network $G(V,E)$, the link quality $q_{u,v}$ of each
    link $(u,v) \in E$, and the node PCR threshold $\kappa$.
Output: The sniffer set $S$ that meets the requirement of minimum-
    $\kappa$-covered for each node.
 1: procedure SelectSniffers($G, \kappa$)
 2:     $S \leftarrow \emptyset$,  $CoveredSet \leftarrow \emptyset$
 3:     $CandidateSet \leftarrow V$
 4:     while $CoveredSet \neq V$ do
 5:         $maxNewCovered \leftarrow 0$, $newsniffer \leftarrow 0$
 6:         for $x$ in $CandidateSet$ do
 7:             $newS \leftarrow S \cup \{x\}$
 8:             $xCovered \leftarrow G$.getCoveredSet($newS, \kappa$) $-$
 9:                         $CoveredSet$
10:             if $|xCovered| > maxNewCovered$ then
11:                 $maxNewCovered \leftarrow |newCovered|$
12:                 $newsniffer \leftarrow x$
13:                 $newCovered \leftarrow xCovered$
14:         $S \leftarrow S \cup \{newsniffer\}$
15:         $CoveredSet \leftarrow CoveredSet \cup newCovered$
16:         $CandidateSet \leftarrow CandidateSet$ - $S$
17:     return $S$
```

Practical sniffer deployment. The sniffer can be deployed in two ways. First, the sniffer can be deployed as software. For example, the node can enable the promiscuous mode to monitor the traffic of its neighbors without affecting its regular data traffic. This is an appropriate way for powerful nodes like Imote2. The second way is to deploy external nodes as sniffers. This is appropriate for less powerful nodes like TelosB. In our current implementation of NetVison on the WSN, we deploy sniffers as software by enabling the promiscuous mode on selected nodes.

Trace recording. NetVision collects packet transmission information from its header at each hop and the arrival time. To further reduce the trace size, NetVision enables the user to specify packet types of interest. For example, a network runs an application which uses the CTP protocol to collect sensor data and also uses an FTSP protocol for global time synchronization. We can set the target type as "CTP" to monitor the data packets and ignore the "FTSP" packets and the acknowledgment packets.

Trace collection. A simple way of retrieving traces is via the serial connection. However, it is only possible in an indoor testbed. In outdoor deployment, most sniffers may not be easily accessible. For this circumstance, we can employ a mobile agent to collect traces such as Tinybee [22]. By doing so, NetVision can avoid large traffic overhead and yet, has no impact on the regular traffic.

Trace inference. The sniffers could miss some packets and, consequently, result in incomplete packet trajectories. For an incomplete packet trajectory, we first explore the trajectories of packets originated from the same source node to find whether there is a packet following the same path. If found any, i.e., its transmission information at each hop is exactly the same as that of the incomplete trajectory, we assume it follows the same path as the packet of the incomplete trajectory, and use its trajectories to infer the missing packet trajectories. If not found, we explore the packets recently sent by the node where the packet trajectory missed. By using this simple inference strategy, the packet capture ratio increases 8% in out testbed experiment on average.

VI. APPLICATIONS

The final goal is to facilitate easy measurement. NetVision provides simple instructions that can support various measurement or debugging applications. In this section, we first give an overview of the NetVision instructions and then provide several examples of using them for measurement or debugging network problems.

A. NetVision Instructions

We assume that each node is assigned a unique node ID which is represented as `nodeID`. A `linkID` is a pair of adjacent `nodeIDs` ($< v_i, v_j >$). A `Path` is a list of `nodeIDs` ($< v_i, v_j, ... >$). A `packetID` is a 2-tuple ID ($<$ `Origin`, `SeqNo` $>$) where `Origin` refers to the node ID of the origin node generating the packet and `SeqNo` represents the sequence number of the packet. A `timeRange` is defined as a pair of timestamps ($< t_i, t_j >$).

NetVision provides three simple instructions and a set of built-in APIs:

- `TRACE`: Specify the protocol type (`protocol`) or other properties to filter packets of interest on the sniffer. For example, `TRACE protocol == CTP` means NetVision only monitors CTP packets [23]. By default, NetVsion monitors all types of network traffics. Other properties include the origin node ID (`Origin`) and the forwarder ID (`FwdID`).
- `CFIND`: Conditionally find the metrics user specified which can be: nodeID, pktID, link, and path. The conditions are specified using the key word `WHERE`. Specifically, `AND` is used to combine multiple conditions and `IN` specifies the desired time range `timeRange`.
- `CEXEC`: Conditionally execute the subsequent instructions.
- Built-in APIs: A set of built-in APIs that perform the basic counting or reading operations on the trace. Specifically, `getCount(nodeID)` returns the appearance times of `nodeID` in the collected trace; `getPRR(nodeID)` returns the packet reception ratio of `nodeID` at sink; `getPDR(link)` returns the packet delivery ratio of link; `getPath(packetID)` returns the `Path` traveled by `packetID`; `getLasthop(packetID)` returns the `nodeID` of `packetID`'s last hop.

B. Measurement Applications

Hotspots analysis. Consider the demand of monitoring node hotspots within the network which is a basic need for network monitoring. A hotspot is a node that appears to be the source of, or destination of, more packets than others. Hotspots can cause congestions or packet losses. Thus, it is important to localize hotspots for network monitoring or diagnose. However, with traditional in-band techniques, it is not easy to localize hotspots from the sink's view, since the packets sent/forwarded by the hotspots may fail to reach the sink in the end.

NetVision can provide fine-grained per-packet per-hop transmission visibility inside the network. With a merged fine-grained trace, we can find `nodeIDs` which appear more than the threshold (e.g., 2000) in a given time range (i.e., *tRange*). These nodes are regarded as hotspots. This application can be simply written as follows:

```
CFIND nodeID
WHERE getCount(nodeID) > 2000 IN tRange
```

Link loss ratio measurement. Understanding the wireless link performance is very helpful for both protocol designers and network managers. A wireless link is usually characterized by its loss ratio. Existing works usually infer the link loss ratio based on network tomography approaches which target at static or slowly changing routing paths and are not suitable for dynamic wireless networks. To cope with the problem in dynamic multihop wireless networks, the state-of-art method Dophy [24] directly encodes the number of retransmissions into the packet at each hop and decode them at the sink for link loss ratio inference. However, it can be inaccurate in poor network condition where packet loss occurs frequently. Different from them, NetVision can measure the link loss ratio accurately based on the actual packet transmissions even in poor network condition. Specifically, `getPDR(linkID)` is used to return the packet delivery ratio of a specified link.

Packet loss localization. One way to debug poor performance due to high packet loss rate of certain nodes is to figure out where packets are dropped in transit. Packet loss localization is hard for in-band techniques due to their limitation to observe packet receptions only at the sink. Using NetVision, we can observe the packet transmissions from multiple vantage points in the network and reconstruct the trajectory of each packet. The following example shows how to localize the packet loss. It first finds out lossy nodes whose packet reception rate is lower than a threshold (e.g., 60%). Then, missing packets originated from lossy nodes can be obtained as follows. For each missing packet, we regard the last hop on its path as the suspicious location and count the number of missing packets at each suspicious location. Finally, the result is a count table where each entry denotes the node that has missing packets.

```
CFIND nodeID
WHERE getPRR(nodeID) < 60% IN tRange
CEXEC{
    CFIND packetID
    WHERE packetID.Origin==nodeID AND \
          getLasthop(packetID)!=SINK
    CEXEC{
       susp_node = getLasthop(packetID)
       susp_cnt[susp_node] += 1 }}
```

Within just ten lines, NetVision helps locate the suspicious packet loss for the operators.

VII. EVALUATION

Previously, we have shown the generality of NetVision by implementing a set of measurement or debugging applications. In this section, we evaluate the performance of NetVision both in extensive simulations and a real indoor testbed.

A. Experimental Setup

Network settings. For simulations, we generate network topologies by the default topology generation tool in TinyOS. We generate grid topologies with 5×5, 8×8, 15×15, 20×20 nodes with $1m$ inter-node spacing. For the testbed experiments, our in-lab testbed which includes 80 TelosB nodes in a 8×10 grid. We consider periodic networks that each node generates and transmits packets to the sink with a period of 1 minute. In each configuration, the sink node is placed at one corner. Each node employs the CTP protocol [23] for transmitting and forwarding packets. The node PCR threshold κ is set to be 0.75 by default.

Sniffer settings. In our implementation, we use ordinary nodes as sniffers. They are selected from the network based on our sniffer deployment strategy. To turn an ordinary node into a sniffer, we need to disable its hardware address recognition. By doing this, the node will not reject packets that are not destined to it and thus can be used as a sniffer to monitor its neighborhood. The sniffer node still serves as a common node that forwards packets destined to it. The sniffers then transfer traces directly to a PC via serial connections.

B. Main Results

We evaluate NetVision's measurement accuracy by applying it to a real case. We choose to measure link loss ratio in this experiment since it is an essential and common measurement task in real network management. The sniffer deployment strategy we employ here is link-correlation-aware-sniffer-deployment. For comparison, we also evaluate the accuracy of a typical in-band technology Dophy [24] and an out-band technology SMSN [14]. Dophy is the state-of-art in-band work on link loss ratio measurement in WSN. It encodes the number of retransmissions at each hop into the packet and recovers them at the sink for link loss ratio estimation. SMSN is a typical sniffer deployment strategy, and we extend it with our trace merging scheme to perform loss rate measurement. We refer to it as Enhanced-SMSN.

We run them in a simulated 225-node network and compare the measurement accuracy in terms of RMSE (root-mean-square error) under different link qualities. To simulate correlated links, neighbors of a node selectively to drop the same packets from that node. As shown in Fig. 3, NetVision has the RMSE always below 10% and consistently achieves the highest accuracy under all scenarios. Specifically, NetVision improves accuracy significantly by 63% on average compared to Dophy and 27% compared to Enhanced-SMSN. The reason is that with loss links (< 0.5), Dophy would miss many retransmission information due to: (1) a significant percent of packets are dropped in flight, (2) many packets change their next hop and update the number of retransmissions at that hop. Enhanced-SMSN deploys sniffers only based on the topology and remains the same under all scenarios. Therefore, more packets would be missed under lower link qualities. On the contrary, NetVision can adapt to various network conditions and ensure the packet capture ratio of each node by considering the link quality and link correlation.

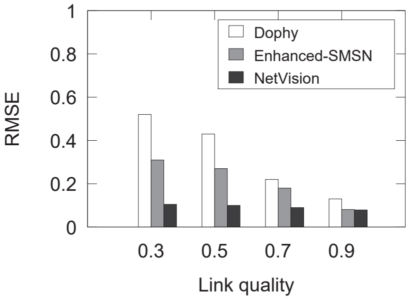
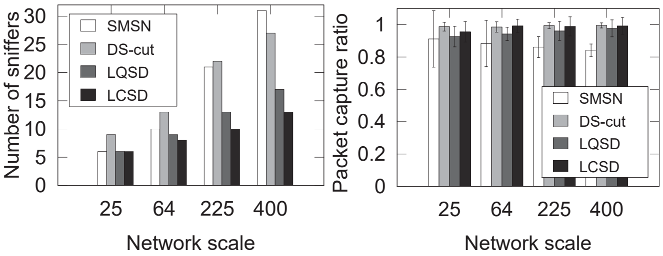


Fig. 3: Measurement accuracy on link loss ratio.



(a) Sniffers.
(b) Packet capture ratio.

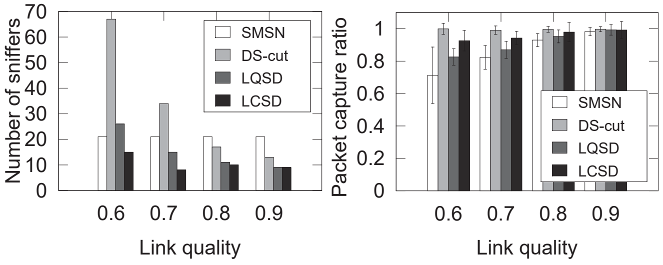
Fig. 4: Impact of network scale.

C. System Insights

In this set of experiments, we evaluate the performance of our sniffer deployment strategies. In addition to our link-quality-aware-sniffer-deployment strategy (**LQSD**) and link-correlation-aware-sniffer-deployment strategy (**LCSD**), we implement two other algorithms for comparison.

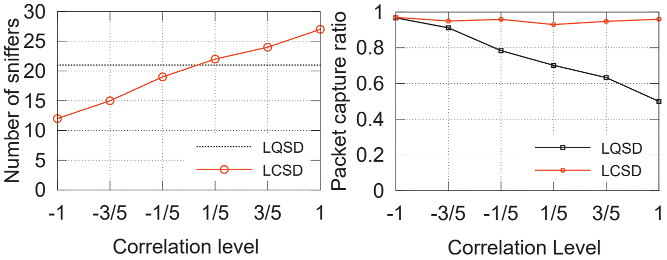
SMSN:We implement SMSN [14] in our scenario. The key idea is to minimize the number of sniffers to monitor all communication links.

DS-cut: It first deletes all the links whose link quality is less than the PCR threshold κ. Then, it greedily selects a minimum dominating set as sniffer set.



(a) Sniffers. (b) Packet capture ratio.

Fig. 5: Impact of link quality.



(a) Deployment cost. (b) Packet capture ratio.

Fig. 6: Impact of link correlation.

Impact of network scale: Fig. 4 shows the performance of different algorithms in terms of the number of sniffers and average packet capture ratio (PCR). From Fig. 4(a), we can see that: (1) The number of sniffers of LQSD and LCSD is consistently smaller than the other two algorithms. (2) The number of sniffers needed by DS-cut and SMSN increases quickly with increasing network scales. This is due to that LQSD and LCSD can cover nodes through multiple links with link quality less than κ. However, DS-cut cuts all these links and thus needs to deploy more sniffers. Different from the other three algorithms, SMSN aims at monitoring all links which can exponentially grow in the amount with the increasing network scales.

From Fig. 4(b), we can see that: (1) Both LQSD and LCSD can achieve high PCR, i.e., PCR>90%. (2) The PCR of SMSN is smaller than the other three algorithms. The reason is that SMSN only ensures there is a link connecting the node and sniffer. The PCR of the node can be very low with a low-quality link. (3) DS-cut always achieves the highest PCR since it cuts off all the low-quality links. The above results show that LQSD and LCSD are highly scalable.

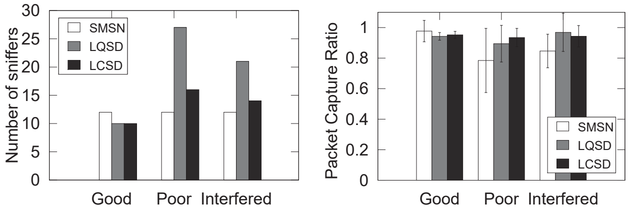
Impact of link quality: From Fig. 5(a) and Fig. 5(b), we can see that: (1) When the link quality is high (PRR=0.9), all of them can achieve high PCR with a small number of sniffers. Note that, all algorithms except SMSN need fewer than 12 sniffers for a 225-node network. This is because SMSN selects sniffers only based on the topology. (2) When link quality is relatively low (PRR=0.6), the sniffer number of DS-cut is significantly higher than other algorithms. This is because many low-quality links are cut. (3) The sniffer number of LCSD is much smaller than LQSD when the link quality is low. Specifically, LCSD needs 42% fewer sniffers than LQSD under link quality 0.6. This is because link correlations can have a larger impact on the link performance when the link quality is low. The above results show that LCSD is robust even under low link qualities.

Impact of link correlation: To focus on the effect of link correlation, link quality is fixed at 0.5. We evaluate the performance of our sniffer deployment strategy under different correlation levels. We take the concept of correlation level from TWC15 [21]. Correlation level 1 represents that links related to the same node are perfect positively correlated (i.e., their packet reception statuses are exactly the same). Similarly, -1 represents those links are perfect negatively correlated, and 0 represents links are independent. We generate the packet reception traces for different link correlation levels. Results are shown in Fig. 6. We can see that:(1) The number of sniffers needed by LCSD increases when the correlation level grows. This is due to that multiple sniffers can jointly cover one node by exploiting the diversity of the packet receptions when the correlation level is low. When the packet receptions of sniffers around a node are highly related, they are likely to miss same packets from the node, which results in more sniffers needed to be deployed. (2) When correlation level below 0, LCSD needs to deploy fewer sniffers while achieving higher PCR. Specifically, LCSD deploys 40% fewer sniffers and achieves 29.3% higher PCR when links are perfect negatively correlated. The reason is that LCSD exploits the packet reception diversity with negatively correlated sniffers. (3) When the correlation above 0, LCSD needs to deploy more sniffers and achieves higher PCR. The reason is that LQSD overestimates the real reception ratio of multiple links.

D. Testbed Study

We implement NetVision on the Telosb/TinyOS platform in an indoor laboratory. We adopt interferences to introduce different link conditions. We evaluate NetVison under three different kinds of link conditions. *Good links:* The average packet reception rate (PRR) of each link is larger than 90%. *Poor links:* The average PRR is less than 30%. *Interfered links:* We employ WiFi interference to further generate a different degree of link correlation. In the presence of WiFi interference, the TelosB nodes are placed closely to a laptop and a wireless access point with the 802.11g mode. We use iperf [25] to generate ~5MB WiFi traffic.

We run LQSD, LCSD, and SMSN. We measure their deployment cost under different link conditions, and results are shown in Fig. 7. Under good link conditions, all of them show high PCR with a small number of sniffers needed. This is because the link quality and link correlation contribute little to the sniffer selection with good link conditions. Under poor link conditions, SMSN deploys the fewest sniffers since it only depends on the network topology. However, the PCR of SMSN is the lowest as shown in Fig. 7(b). LCSD deploys 37% fewer sniffers than LQSD while achieving 14% higher PCR. Under interfered link conditions, we observe that the



(a) Deployment cost. (b) Packet capture ratio.

Fig. 7: Results on testbed study.

performance gain of LCSD is still significant, and the gain is lower than that under the bad link conditions. This is because that sniffers around the inference source have similar packet reception patterns and are positively correlated.

To evaluate the accuracy of NetVision's synchronization strategy, we deploy another sensor to broadcast packets with transmission power 0dBm to ensure every sniffer can hear. Since the transmission time can be ignored, the broadcast packet is regard to arriving at each sniffer at the same time. We use the synchronization tree built by NetVision to synchronize the packet arriving time at each sniffer to that at the root. The packet arriving time at the root is the ground truth. The average synchronization error for every 15 min over 1h is 0.029ms. The result shows that our synchronization scheme achieves high accuracy.

VIII. CONCLUSION

This paper presents NetVision, a practical network measurement system with special consideration for sniffer deployment. The key insight of NetVision's sniffer deployment algorithm is to explicitly consider the link quality and link correlation between network nodes and sniffer nodes, with the assumption that sniffer nodes can only be deployed on positions of network nodes. As such, NetVision can accurately quantify the packet capture ratio of each node for a given sniffer deployment. We then formulate the sniffer deployment problem as an optimization problem to minimize the number of sniffers while satisfying the packet capture ratio for each node. We propose efficient heuristic algorithms for this problem; and further design a set of instructions and APIs to simplify the development and implementation of a variety of measurement tasks. We implement and evaluate NetVision extensively using simulations and an indoor testbed. Results show that NetVision is generic, robust, and effective.

ACKNOWLEDGMENT

This work is supported by the National Science Foundation of China (No. 61772465, 61502417, 61472360), Alibaba-Zhejiang University Joint Institute of Frontier Technologies, Zhejiang Provincial Key Research and Development Program (No. 2017C02044), and the Fundamental Research Funds for the Central Universities (No. 2017FZA5013). This work was also supported by a Canada NSERC Discovery Grant, an NSERC E.W.R. Steacie Memorial Fellowship, and a MITACS project grant. Wei Dong is the corresponding author.

REFERENCES

[1] J. Wang, Z. Cao, X. Mao, X.-Y. Li, and Y. Liu, "Towards energy efficient duty-cycled networks: Analysis, implications and improvement," *IEEE Trans. on Computers*, vol. 65, no. 1, pp. 270–280, 2016.

[2] T. Winter, "Rpl: Ipv6 routing protocol for low-power and lossy networks," *RFC 6550*, 2012.

[3] T. Lee, M. S. Lee, H. S. Kim, and S. Bahk, "A synergistic architecture for rpl over ble," in *Proc. of IEEE SECON*, 2016.

[4] S. Zhang, G. Zhang, A. Yan, Z. Xiang, and T. Ma, "A highly reliable link scheduling strategy for wirelesshart networks," in *Proc. of IEEE ATC*, 2013.

[5] "Buetooth SIG. bluetooth sig 2016 technology roadmap," https://www.bluetooth.com/news/pressreleases/2015/11/11/bluetooth-technology-to-gain-longer-range-faster-speed-mesh-networking-in-2016, accessed: 2017-07-26.

[6] Y. Liu, K. Liu, and M. Li, "Passive diagnosis for wireless sensor networks," *IEEE/ACM Trans on Networking*, vol. 18, no. 4, pp. 1132–1144, 2010.

[7] Y. Gao, W. Dong, C. Chen, X. Zhang, J. Bu, and X. Liu, "Accurate per-packet delay tomography in wireless ad hoc networks," *IEEE/ACM Trans. on Networking (TON)*, vol. 25, no. 1, pp. 480–491, 2017.

[8] Y.-C. Cheng, J. Bellardo, P. Benkö, A. C. Snoeren, G. M. Voelker, and S. Savage, "Jigsaw: Solving the puzzle of enterprise 802.11 analysis," in *Proc. of ACM SIGCOMM*, 2006.

[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*, 2006.

[10] N. Ramanathan, K. Chang, R. Kapur, L. Girod, E. Kohler, and D. Estrin, "Sympathy for the sensor network debugger," in *Proc. of ACM SenSys*, 2005.

[11] Q. Ma, K. Liu, X. Miao, and Y. Liu, "Sherlock is around: Detecting network failures with local evidence fusion," *IEEE Trans. on Parallel and Distributed Systems*, vol. 26, no. 5, pp. 1430–1440, 2015.

[12] B.-R. Chen, G. Peterson, G. Mainland, and M. Welsh, "Livenet: Using passive monitoring to reconstruct sensor network dynamics," in *Proc. of IEEE DCOSS*, 2008.

[13] W. Dong, Y. Liu, Y. He, T. Zhu, and C. Chen, "Measurement and analysis on the packet delivery performance in a large-scale sensor network," *IEEE/ACM Trans on Networking*, vol. 22, no. 6, pp. 1952–1963, 2014.

[14] D. Dong, X. Liao, Y. Liu, C. Shen, and X. Wang, "Edge self-monitoring for wireless sensor networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 22, no. 3, pp. 514–527, 2011.

[15] W. Zeng, X. Chen, Y.-A. Kim, Z. Bu, W. Wei, B. Wang, and Z. J. Shi, "Delay monitoring for wireless sensor networks: An architecture using air sniffers," in *Proc. of IEEE MILCOM*, 2009.

[16] S. Wang, A. Basalamah, S. M. Kim, G. Tan, Y. Liu, and T. He, "A unified metric for correlated diversity in wireless networks," *IEEE Trans. on Wireless Communications*, vol. 15, no. 9, pp. 6215–6227, 2016.

[17] S. Wang, S. M. Kim, Y. Liu, G. Tan, and T. He, "Corlayer: A transparent link correlation layer for energy efficient broadcast," in *Proc. of ACM MobiCom*, 2013.

[18] R. Zheng, T. Le, and Z. Han, "Approximate online learning for passive monitoring of multi-channel wireless networks," in *Proc. of IEEE INFOCOM*, 2013.

[19] K. Zeng, Z. Yang, and W. Lou, "Opportunistic routing in multi-radio multi-channel multi-hop wireless networks," in *Proc. of IEEE INFOCOM*, 2010.

[20] "Network measurement in multihop wireless networks with lossy and correlated links [technical report]," August 2017. [Online]. Available: https://www.dropbox.com/s/b68f2gcjuw7fg9p/netvision-tech.pdf

[21] S. Wang, A. Basalamah, S. M. Kim, S. Guo, Y. Tobe, and T. He, "Link-correlation-aware opportunistic routing in wireless networks," *IEEE Trans. on Wireless Communications*, vol. 14, no. 1, pp. 47–56, 2015.

[22] K. Ota, M. Dong, and X. Li, "Tinybee: mobile-agent-based data gathering system in wireless sensor networks," in *Proc. of IEEE NAS*, 2009.

[23] O. Gnawali, R. Fonseca, K. Jamieson, M. Kazandjieva, D. Moss, and P. Levis, "CTP: An Efficient, Robust, and Reliable Collection Tree Protocol for Wireless Sensor Networks," *ACM Trans. on Sensor Networks*, vol. 10, no. 1, pp. 16:1–16:49, 2013.

[24] C. Cao, Y. Gao, W. Dong, and J. Bu, "Fine-grained loss tomography in dynamic sensor networks," in *Proc. of IEEE ICPP*, 2015.

[25] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs, "Iperf: The tcp/udp bandwidth measurement tool," *http://dast. nlanr. net/Projects*, 2005.