# ProgLIMI: Programmable LInk Metric Identification in Software-Defined Networks

Xiong Wang, *Member, IEEE*, Mehdi Malboubi, Zhihao Pan, Jing Ren, Sheng Wang, Shizhong Xu, and Chen-Nee Chuah, *Fellow, IEEE*

***Abstract*— In this paper, we propose the Programmable LInk Metric Identification (ProgLIMI) infrastructure for software-defined networking (SDN) networks. ProgLIMI identifies round-trip link metrics (RTLMs) from accumulated end-to-end metrics of selected measurement paths by leveraging the flexible routing control capability of SDN networks. ProgLIMI mainly solves three sub-problems: 1) monitor placement; 2) linearly independent measurement path construction; and 3) flow rule design. To reduce measurement cost, ProgLIMI tries to minimize the number of required monitors and flow rules. In this paper, we address the three sub-problems for both full and hybrid SDN networks. For full SDN networks, ProgLIMI can achieve full RTLM identification using only one monitor and two flow rules in each SDN switch. In contrast, the RTLM identification in hybrid SDN networks is more complicated due to the routing constraint of hybrid SDN networks. We first prove that the monitor placement problem in hybrid SDN networks is NP-hard. We then formulate the monitor placement and measurement path selection problem in hybrid SDN networks and propose a greedy heuristic algorithm to solve the problem efficiently. Our evaluations on both physical testbed and simulation platform reveal that ProgLIMI can accurately identify the RTLMs (delay and loss rate). Besides, ProgLIMI is also resource efficient, i.e., it only requires two flow rules in each SDN switch and a small number of monitors, and the extra probing traffic load incurred by ProgLIMI is also low.**

***Index Terms*— Software-defined networking, link metric identification, round-trip link metric, monitor, measurement path.**

## I. Introduction

ACCURATE and timely monitoring of time-varying link performance metrics (e.g., delays, loss rates, and available bandwidth) are essential for various daily SDN network management tasks such as performance diagnosis, traffic engineering, and resource allocation, which improve the performance and resource utilization efficiency of SDN networks. Generally, two broad approaches can be used to measure link metrics: direct and indirect approaches. The direct approaches directly measure the status of the ports (links) at each node (router or switch). While the indirect approaches infer link metrics/states by measuring the performance of selected paths between monitors, which are connected to a subset of nodes.

The direct approaches have many potential limitations. First of all, not all network elements have the performance monitoring capability or such capability may be disabled due to resource limitations (e.g., CPU load, Memory). Second, monitoring link metrics at each node and gathering measurement data from all nodes are not scalable in large-scale networks. Lastly, in-node measurements, such as the SNMP statistics, may not be trustworthy since the real network operation experience show that switches may drop packets even though their SNMP statistics show everything is fine [1].

Hop-by-hop direct measurement approaches use diagnostic tools such as ping, traceroute, pathchar, and clink to measure the performance metrics of hop-by-hop links on probe packets forwarding paths. These diagnostic tools measure link performance metrics by exchanging Internet Control Message Protocol (ICMP) packets with each intermediate node. However, for network security concerns, the ICMP is disabled in some intermediate nodes, which makes these tools infeasible to use in production networks. Moreover, per-hop measurement incurs high traffic load that may lead to increased congestion.

To mitigate the measurement overhead, end-to-end approaches provide a light-weight solution. Instead of per-hop measurement, end-to-end approaches measure the accumulated end-to-end metrics of a set of selected paths and use the *network tomography* technique [2] to infer the performance metrics of individual links from the end-to-end measurements. Since only a small number of end-to-end measurements are required, end-to-end approaches eliminate the dependence on the cooperation of network elements on the measurement paths and reduce the extra probing traffic overhead significantly [2].

Generally, the end-to-end approaches can be used to identify broad types of link metrics, such as delay, packet loss rate, throughput, and available bandwidth, etc. We observed that the performance metrics (e.g., link delay) of most interest to both users and network providers are usually additive. That is, the end-to-end metric over a path of multiple links is the sum of individual link metrics. Thus, this paper only considers

Manuscript received September 11, 2016; revised April 29, 2017, November 22, 2017, and June 5, 2018; accepted August 2, 2018; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor K. Argyraki. Date of publication September 10, 2018; date of current version October 15, 2018. This work was supported in part by the NSFC Funds under Grant 61671130, Grant 61271165, and Grant 61301153, in part by the National Basic Research Program (China's 973 Program) under Grant 2013CB329103, and in part by the Fundamental Research Funds for the Central Universities under Grant ZYGX2016J002. *(Corresponding author: Xiong Wang.)*

X. Wang, J. Ren, S. Wang, and S. Xu are with the Key Laboratory of Optical Fiber Sensing and Communications, School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: wangxiong@uestc.edu.cn).

M. Malboubi was with the Department of Electrical and Computer Engineering, University of California at Davis, Davis, CA 95616 USA. He resides in San Ramon, CA 94582 USA.

Z. Pan is with the NETEASE, Inc., Guangzhou 510000, China.

C.-N. Chuah is with the Department of Electrical and Computer Engineering, University of California at Davis, Davis, CA 95616 USA.

Digital Object Identifier 10.1109/TNET.2018.2865892

how to identify additive link metrics from end-to-end path measurements. Since a multiplicative metric (e.g., packet loss rate) can be expressed in an additive form by using the $\log(\cdot)$ function, we can treat the additive and multiplicative metrics equivalently.

If link metrics are additive, the problem of identifying link metrics using end-to-end path measurements can be formulated as a system of linear equations $\mathbf{AX} = \mathbf{Y}$, where $\mathbf{X}$ is the vector of unknown link metrics, $\mathbf{A}$ is a binary routing matrix for the selected end-to-end measurement paths, and $\mathbf{Y}$ is the vector of end-to-end path measurements. Each element of $\mathbf{Y}$ equals to the sum of the corresponding link metrics. Accordingly, given $\mathbf{Y}$ and $\mathbf{A}$, the link metric identification problem is defined as the process of estimating unknown vector $\mathbf{X}$.

Currently, there are two main categories of approaches for solving the link metric identification problem: statistical and algebraic approaches. The statistical approaches [3], [4] assume that the link metrics follow some probability distributions, and use various statistical inference techniques to estimate the link metric distributions from the measured path metrics. Different from statistical approaches, algebraic approaches assume that the link metrics are constants in small timescale, and use linear algebraic methods to calculate link metrics from the measured metrics of a set of end-to-end measurement paths. Compared with statistical approaches, the algebraic approaches can accurately and uniquely identify the link metrics. Therefore, this paper uses the algebraic approach to identify round-trip link metrics for SDN networks.

From the viewpoint of linear algebra, the linear system $\mathbf{AX} = \mathbf{Y}$ has a deterministic solution if and only if the coefficient matrix $\mathbf{A}$ is invertible, i.e., the number of linearly independent row vectors in $\mathbf{A}$ must equal to the number of unknown variables in vector $\mathbf{X}$. Thus, to uniquely identify all link metrics, we must first deploy $m$ linearly independent measurement paths in a network, where $m$ is the number of links. Furthermore, it is clear that a measurement path must start and end at monitors, which are used to send, receive, and process probe packets. Thus, to ensure that there exists a sufficient number of linearly independent paths in a network, a certain amount of monitors are needed to be placed in the network, and to save cost, the number of monitors required should be minimized.

In today's IP networks, the link metric identification problem mainly faces the following challenges. First, the link metric identification needs to establish a set of linearly independent paths between the monitors, but due to lack of flexible control plane, traditional IP networks mostly employ the shortest path routing, and the explicit routing is not allowed. Although MPLS or source routing protocols can be used to set up required measurement paths, the maintenance cost is very high. Second, under the routing constraint, a large number of monitors may be required to meet the full identifiability condition [5], which increases the cost and may even be infeasible due to the placement constraints (e.g., some core nodes are prohibited to place monitors).

The emergence of SDN brings new opportunities to overcome the challenges faced by the link metric identification problem efficiently. SDN is an emerging networking architecture, which separates the control plane and the data plane, and the control plane has the programmable capability to control the forwarding configurations in data plane running in each switch. Consequently, SDN allows for more complex and flexible network measurement and management tasks. Specifically, for the link metric identification problem, we can easily deploy various measurement paths (simple paths, multicast trees, or paths with loops) in SDN networks. Thus, with the help of SDN, we can achieve full link metric identifiability by using a small number of monitors. Besides monitors, link metric identification in SDN networks needs that TCAM entries correctly define and implement forwarding rules for measurement paths. Also, note that probe packets transmitted on measurement paths consume link bandwidth. So to reduce the network resource consumption, the measurement paths and forwarding rules for probe packets should be carefully designed.

In this paper, we investigate the Round-Trip Link Metric (RTLM) identification problem in SDN networks. More specifically, we seek to address the following three questions: i) For a given SDN network, how we can place monitors to achieve full RTLM identifiability. ii) Given a placement of monitors, how we can construct the linearly independent measurement paths between monitors. iii) Given the measurement paths, how we can design the forwarding rules (installed in TCAM) to realize the measurement paths. To reduce the measurement cost, we aim to minimize the number of required monitors and TCAM entries, and the total length of the measurement paths. In this paper, we assume that the link metrics are additive and change slowly relative to the measurement process (i.e., the link metrics can be viewed as constants).

## II. Related Works and Our Contributions

### A. Related Works

The existing works on link metric identification problem usually model the metric of a link as either a random variable or a constant. For the random model, the statistical approaches [3], [4], [6]–[8] are used to estimate the distributions or parameters (e.g., variances and mean) of the random link metrics from realizations of path metrics. While for the constant model, as in this paper, the algebraic approaches [4], [9], [11]–[15] are used to compute link metrics from path measurements.

Depending on whether the distributions of link metrics are known or not, the random model can be further classified into parametric models and nonparametric models. Duffield and Lo Presti [3] and Shih and Hero [6] investigate the link delay inference problem under the nonparametric model. In [3], the variances of the link delays are estimated using a moment method. In [6], the objective is to infer link delay distributions from end-to-end measurements. Parametric models are considered in [4] and [7], where the link delay distribution is from a known parametric family, such as the exponential family or Gaussian family, and statistical techniques are applied to infer the parameters of the link delay distribution. In the end-to-end metric measuring process, both multicast trees and unicast paths

can be exploited to carry probe packets. Multicast-based measurement [3], [7], [8] incurs low overhead, but it is not widely allowed in practical networks. In contrast, although unicast-based measurement [4], [6] incurs higher overhead, it is more practical and flexible.

Unlike statistical approaches, the algebraic approaches can get an accurate and deterministic solution for the link metric identification problem. In recent years, the algebraic approaches attracted more attentions [4], [9], [11]–[15]. Xu *et al.* [9] prove that a link metric vector with no more than $k$ nonzero elements can be recovered from $O(k \cdot log(n))$ (where $n$ is the number of network nodes) path measurements using compressive sensing. However, the compressive sensing based method [9] cannot be used for networks with arbitrary valued link metrics. Gopalan and Ramasubramanian [11] give the necessary and sufficient conditions to identify link metrics by using cycles and proposes an efficient algorithm to construct linearly independent measurement cycles or paths containing cycles. The same authors also propose an algorithm to find the maximum number of linearly independent paths/cycles that can be constructed between the given monitors [12]. Since routing along cycles is always prohibited by routing protocols, Ma *et al.* [13] gives the necessary and sufficient conditions to identify link metrics by using simple paths (paths without cycles), and develop a monitor placement algorithm based on these conditions. Ma *et al.* [14] propose an efficient algorithm to construct linearly independent measurement paths. In addition, Ma *et al.* [15] also investigate the monitor placement problem for achieving maximum network identifiability when the number of monitors is limited.

To get external measurements, the afore-mentioned work [4], [9], [11]–[15] must rely on the explicit routing techniques (e.g., source routing or MPLS) to establish measurement paths. This requirement may not be satisfied in practical networks since most IP networks adopt the shortest path routing paradigm, and most of the routers do not support the explicit routing [16]. Thus in [5], we address the end-to-end path based link metric identification problem under the shortest path constraint. Our study shows that with the shortest path routing constraint, the number of monitors required can be very large.

In addition, all of the work in [3]–[9] and [11]–[15] assume that the link metrics are symmetric, i.e., the metrics of the links in two directions are the same. However, this assumption usually does not hold in practical networks due to asymmetric loads on the links in two different directions or different packet scheduling strategies [10]. Thus, the one-way link metric identification problem in directed networks (links in different directions have different metrics) is studied in [4] and [10]. Gurewitz and Sidi [10] show that directed network is unidentifiable if we only use cyclic measurement paths starting and ending at the same monitor. Furthermore, Xia and Tse [4] prove that a directed network is identifiable only if every non-isolated node is a monitor. To obtain one-way link delays, Gurewitz and Sidi [10] and Firooz and Roy [38] estimate the one-way delays from cyclic-path delay measurements. However, the estimated one-way delays may have errors.

From the existing studies, we can make the following two observations: 1) Due to the lacking of flexible routing control capability, constructing linearly measurement paths is a tough challenge in traditional IP networks; 2) Accurately identifying one-way link metrics is costly or even infeasible for practical networks. Thus, in this paper, we study how to efficiently and accurately identify RTLM from end-to-end measurements in SDN networks. As we know, SDN has powerful centralized and programmable control plane, which can support various measurement paths, such as simple paths, paths with cycles, and multicast trees. Thus, the emergence of SDN paves the way for achieving low-cost and efficient link metric identification. In this paper, we study the RTLM identification problem in SDN networks. To the best of our knowledge, only a few work [17]–[20] has addressed the performance metric monitoring in SDN networks. van Adrichem *et al.* [17] and Phemius and Bouet [18] propose to measure the end-to-end metrics by sending and receiving probe packets from the network controller. But the control channels between the controller and SDN switches may be congested, making it prohibitive to get accurate results. More recently, Shibuya *et al.* [19] and Atary and Bremler-Barr [20] propose schemes for monitoring link and round-trip delays in SDN networks, respectively. However, the schemes proposed in [19] and [20] can only work in full SDN networks. In reality, upgrading all existing legacy devices to SDN-enabled ones poses very high budget and operational burden on network providers. Thus network providers usually choose to incrementally deploy SDN devices in their existing networks [21]. As a result, hybrid SDN architecture is likely to be a long-term solution for the real operational networks. In summary, the work in [19] and [20] is the closest in spirit to ours. However, our work is different from the work in [19] and [20]. Specifically, the problem considered in this paper is more general as we consider how to identify any additive RTLM in both full and hybrid SDN networks and our proposed ProgLIMI requires fewer measurement resources.

### *B. Our Contributions*

Our contributions are summarized as follows.

i) We propose a RTLM identification infrastructure called ProgLIMI for SDN networks. ProgLIMI leverages the programmable capability of SDN networks to facilitate the RTLM identification process.

ii) We address the problem of identifying RTLMs of full SDN networks, where all network nodes are SDN-enabled. Specifically, we construct measurement paths, place monitors, and design probe packets forwarding rules for full SDN networks. The proposed scheme only needs one monitor and two TCAM entries to achieve full RTLM identifiability.

iii) We address the problem of identifying RTLMs of hybrid SDN networks, where only a subset of network nodes are SDN-enabled, and the rest are traditional IP routers. Under the routing constraint in hybrid SDN networks, we first prove that the monitor placement problem in hybrid SDN networks is NP-hard. Then we formulate the monitor placement and measurement path selection

problem in hybrid SDN networks and propose a heuristic algorithm to solve the problem efficiently. Lastly, to realize the selected measurement paths in hybrid SDN networks, we design forwarding rules for the probe packets. By properly designing the forwarding rules, only two TCAM entries are required in each SDN nodes.

iv) We implement a prototype and also conduct simulations in Mininet. The evaluation results verify that ProgLIMI can achieve full RTLM identifiability with high accuracy and incurs low measurement cost for both full and hybrid SDN networks.

## III. Problem Formulation

We model an SDN network as a connected undirected graph $G(V, L)$, where $V$ is the set of nodes and $L$ is the set of links. Let $n = |V|$ and $m = |L|$ denote the number of nodes and links, respectively. Each node $v \in V$ could be either an SDN switch or a legacy IP router. In this paper, we consider two SDN network scenarios: full SDN networks and hybrid SDN networks [21]. In full SDN networks, all nodes are SDN-enabled, while in hybrid SDN networks, only a part of nodes are SDN-enabled and the rest are legacy IP routers. Let $V_{SDN}$ denote the set of SDN nodes and $V_{IP}$ denote the set of IP routers ($V = V_{SDN} \cup V_{IP}$). The SDN switches forward packets according to the rules installed in the flow tables, and IP routers forward packets on the shortest paths determined by the link weights.

Each link $l \in L$ is associated with an unknown round-trip performance metric $x_l$ (e.g., delay and loss rate) and a given routing weight $w_l$. The round-trip performance metric of an undirected link $l = (u, v)$ is defined as the summation of the one-way performance metrics of the two directed links $(u, v)$ and $(v, u)$. Clearly, round-trip link performance metric (RTLM for short) cannot accurately reflect the performance of individual one-way links. However, RTLM is also useful for network operators for the following two reasons: First, RTLMs can be viewed as the upper bounds of one-way link metrics; Second, RTLMs can help network operators to locate the abnormals of nodes and links quickly. In this paper, we assume that the RTLMs are additive and change slowly relative to the measurement process (i.e., the RTLMs can be viewed as constants in the measurement process).

We assume that some nodes in the network can be directly connected to monitors, which can send and receive probe packets. The measurement paths start and end at monitors. It is notable that measurement paths with loops are allowed in SDN networks. Let $\mathbf{X} = (x_1, x_2, , x_m)^T$ denote the vector of RTLMs and $\mathbf{Y} = (y_1, y_2, , y_c)^T$ denote the vector of path measurements, where $c$ is the number of measurement paths. Based on the above assumptions and notations, the RTLM identification problem can be represented by the linear system $\mathbf{AX} = \mathbf{Y}$, where $\mathbf{A}$ is the routing matrix, with each entry $A_{ij} \in \{0, 1\}$ indicating whether link $j$ is on measurement path $i$.

Evidently, to uniquely determine $\mathbf{X}$, $\mathbf{A}$ must have full rank, i.e., $rank(\mathbf{A}) = m$. In other words, we must construct $m$ linearly independent measurement paths between monitors to take the measurement. In ProgLIMI, the monitors are used to send and receive probes and calculate the round-trip metrics of measurement paths based on the statistics of probe packets. The monitors can be implemented on commercial servers with customized networking stack and high-speed network interfaces, and thus a monitor will cost thousands or tens of thousands of dollars. If every node connects to a monitor, $\mathbf{A}$ can be an identity matrix and $\mathbf{X}$ simply equals $\mathbf{Y}$. However, placing a monitor at every node has the following issues. Firstly, although the monitors can be implemented on commercial servers with customized networking stack and high-speed network interfaces, deploying a large number of monitors in large SDN networks also incurs high hardware and maintenance costs [13], [22]. Secondly, not all network nodes are allowed to deploy monitors due to the practical limitations (e.g., available ports, security issues). Thirdly, the network operators are reluctant to set up and maintain a distributed network monitoring infrastructure, which has a large number of monitors. At last, the reliability of RTLM identification system decreases with the increasing number of required monitors. Because a monitor just provides a part of measurements for the linear equation $\mathbf{AX} = \mathbf{Y}$. To uniquely determine the RTLMs, all monitors must work correctly. Therefore, if more monitors are involved in the monitoring system, the reliability of the system will be lower. To avoid using monitors in SDN networks, the existing work [23] lets SDN controller play the role of monitors, i.e., the SDN controller generates and receives probe packets. However, this approach will bring high communication and computation overhead to the controller, which incurs the scalability issue. So to mitigate the above issues in this paper, we minimize the number of monitors placed in the network. In addition, forwarding probe packets requires flow rules that are usually installed on TCAMs with very limited capacity, and probe packets also incur extra traffic load on links. Therefore, to mitigate the impact on the network performance, the number of required TCAM entries for forwarding probe packets and the traffic load incurred by probe packets should also be minimized. In summary, given a full or hybrid SDN networks, the objective of this paper is to compute a placement of the minimum number of monitors in $G(V, L)$ and construct $m$ linearly independent measurement paths between monitors that enables the identification of all RTLMs with the minimum consumption of network resources (TCAM entries and link bandwidth).

## IV. RTLM Identification in Full SDN Networks

Full SDN networks have powerful routing control capability, which can be leveraged to facilitate the RTLM identification process. To save measurement cost and mitigate the impact on network performance, we need to carefully place monitors, construct measurement paths, and design forwarding rules for probe packets. Since monitor placement is closely related to measurement path construction, for ease of presentation, we will first introduce how to construct measurement paths and design forwarding rules, and then we will present how to place monitors such that the probing cost can be minimized.

### A. Measurement Path Construction

We start the description of measurement path construction from tree topologies. Let us first consider a tree $T(V_T, L_T)$

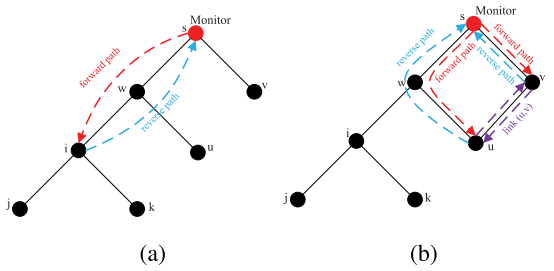



Fig. 1. Measurement path construction for on-tree link and non-tree link. (a) On-tree link. (b) Non-tree link.

rooted at node $s$. As shown in Fig. 1(a), we assume that a monitor is placed at node $s$, and $(i,j) \in L_T$ is an arbitrary link on tree $T(V_T, L_T)$. Let $p_i^f$ and $p_i^r$ denote the forward path and reverse path between nodes $s$ and $i$ on the tree $T(V_T, L_T)$, respectively. To identify the round-trip metric of link $(i,j)$, we can construct two measurement paths $p_i$ and $p_j$, where $p_i = p_i^f \cup p_i^r$ and $p_j = p_j^f \cup p_j^r$. Since $p_i$ and $p_j$ are two circles, the end-to-end measurements ($y_i$ and $y_j$) of $p_i$ and $p_j$ are round-trip metrics of $p_i$ and $p_j$. Based on the measurements $y_i$ and $y_j$ from the constructed paths $p_i$ and $p_j$, we can obtain the round-trip metric of link $(i,j)$ through the following equation:

$$x_{ij} = y_j - y_i \tag{1}$$

From the above procedure, we can observe that the measurement of path $p_i$ is used to compute the round-trip metrics of links incident to node $i$. To uniquely identify all RTLMs on a tree, we need to construct a measurement path $p_i$ for each node $i \in V_T$.

Next, we will discuss how to construct measurement paths for identifying the round-trip metrics of non-tree links, which are defined as the links not on the tree $T(V_T, L_T)$. Let us consider the case in Fig. 1(b), where there is a non-tree link $(u,v)$ between node $u$ and $v$. In this case, we assume that the round-trip metrics of all links on the tree are identified by using the method introduced above. Therefore, if we can construct two measurement paths starting and ending at node $s$ and going through link $(u,v)$ in different directions, the round-trip metric of link $(u,v)$ can be identified. Specifically, the constructed measurement paths for non-tree link $(u,v)$ are $p_{uv} = p_u^f \cup (u,v) \cup p_v^r$ and $p_{vu} = p_v^f \cup (v,u) \cup p_u^r$, where links $(u,v)$ and $(v,u)$ represent the two links in different directions, respectively. To identify the round-trip metric of a non-tree link $(u,v)$, ProgLIMI just needs to measure the round-trip metrics of $p_{uv}$ and $p_{vu}$. Since the round-trip metrics of on-tree links are known, the round-trip metric of non-tree link $(u,v)$ can be easily obtained as follows:

$$x_{uv} = y_{uv} + y_{vu} - \sum_{(i,j) \in p_{uv}, (i,j) \neq (u,v)} x_{ij} \tag{2}$$

where $y_{uv}$ and $y_{vu}$ are the round-trip measurements of paths $p_{uv}$ and $p_{vu}$, respectively, and $x_{ij}$ is the round-trip metric of on-tree link $(i,j)$.

Based on the above method for constructing measurement paths for identifying round-trip metrics of on-tree and non-tree links, we are now ready to construct measurement paths for identifying the RTLMs of general topologies. For a general topology $G(V,L)$, a spanning tree $T(V_T, L_T)$ rooted at monitor node $s$ is first computed, and then the links on the spanning tree (links in $L_T$) are marked as on-tree links and the rest of the links (links in $L \backslash L_T$) are marked as non-tree links. The computed spanning tree is also called as probing tree thereafter. Given the probing tree, the measurement paths are constructed as follows: 1) for each on-tree link $(i,j) \in L_T$, a measurement path $p_{ij} = p_j^f \cup p_j^r$ is constructed, where $p_j^f$ and $p_j^r$ are the forward path and reverse path between nodes $s$ and $j$ on the spanning tree, respectively; 2) for each non-tree link $(u,v)$, two measurement paths $p_{uv} = p_u^f \cup (u,v) \cup p_v^r$ and $p_{vu} = p_v^f \cup (v,u) \cup p_u^r$ is constructed. For convenience, we use a unique integer number $I_{uv}$ to identify measurement path $p_{uv}$. In the path metric measuring process, the probe packets are forwarded transmitted along the constructed measurement paths periodically. So, to save link bandwidth consumed by probe packets, we use the shortest path tree of the monitor node $s$ as the spanning tree $T(V_T, L_T)$ for constructing measurement paths. The detailed procedure of measurement path construction for full SDN networks is shown in **Algorithm 1**.

**Algorithm 1** Construct Measurement Paths for a Full SDN Network

**Input:** Network $G(V,L)$ and a monitor node $s$
**Output:** Measurement paths in the form of routing matrix $A$

$A \leftarrow NULL$
Set all link weights of $G(V,L)$ to 1
Find a shortest path tree $T(V_T, L_T)$ from node $s$
**for** each link $(i,j) \in L_T$ (node $j$ is a child of node $i$ in $T(V_T, L_T)$) **do**
  $p_{ij} \leftarrow p_j^f \cup p_j^r$
  Append $p_{ij}$ to $A$
**end for**
**for** each link $(u,v)$ in $L \backslash L_T$ **do**
  $p_{uv} \leftarrow p_u^f \cup (u,v) \cup p_v^r$
  Append $p_{uv}$ to A
  $p_{vu} \leftarrow p_v^f \cup (v,u) \cup p_u^r$
  Append $p_{vu}$ to A
**end for**
Return A

### B. Flow Rule Design

After constructing the required measurement paths, the next problem is how to design flow rules installed in each SDN switch such that the probe packets can be forwarded correctly along each measurement path. We call the probe packets forwarded along a measurement path as a probing flow. According to the OpenFlow specification [33], up to 12-tuple fields can be used by a flow rule to define a flow, and the packet headers can be modified at any SDN switch by adding actions in its flow rule. In this paper, we use source and destination IP address fields $(sip, dip)$ in the probe packet header to identify a probing flow. We assume that the source IP address and destination IP address of a probe packet generated by monitor node $s$ are $ip_s$ and $ip_d$, respectively.

With the flexibility provided by SDN, a naive way for realizing the probing paths is to install a dedicated flow rule in each SDN switch for each probing flow traversing it. However, this would likely to occupy a larger number of flow entries, which are usually installed in TCAM with very limited capacity. Thus, to save TCAM capacity, the flow rules for probing flows should be carefully designed such that the number of required TCAM entries is minimized. To do that, we can aggregate the flow rules based on the characteristics of probing flows.

As presented in the previous subsection, a measurement path consists of two sub-paths: forward path and reverse path on the tree. For example, the measurement path for on-tree link $(i, j)$ is $p_{ij} = p_j^f \cup p_j^r$. The flows forwarded along the forward paths have the same destination IP address $ip_d$ (the IP address of a virtual node $d$) since the probe packet are generated by monitor node $s$. That is to say that all probing flows carried on forward paths match the flow rule with matching fields $(*, ip_d)$. We also can observe that an SDN switch may forward the probing flows coming from a forward path to different adjacent nodes. For example, in Fig. 1(a), node $i$ will forward probing flows coming from forward path $p_i^f$ to its adjacent nodes $j$, $k$, and $w$. In legacy IP routers, only the flow rules with the same prefixes and forwarding port can be aggregated to one rule. However, SDN switches can add multiple actions to a flow rule. Therefore, SDN switches can aggregate the rules with the same prefix and different actions to one rule with multiple actions, i.e., only one rule is needed to handle the probing flows coming from forward paths at each SDN switch. An SDN switch performs multiple actions to a probe packet coming from the forward path. Specifically, for each adjacent node $j$ of node $i$, the node $i$ adds the actions to its rule with matching fields $(*, ip_d)$ as follows:

1) If node $j$ is a child of node $i$ on the probing tree, forward probe packets to node $j$.
2) If node $j$ is a parent node of node $i$ on the probing tree, set the destination IP address and VLAN id of probe packets to $ip_s$ and $I_{ij}$, respectively, and forward probe packets to node $j$.
3) If node $j$ is neither parent node nor child node of node $i$, set the destination IP address and VLAN id of probe packets to $ip_s$ and $I_{ij}$, respectively, and forward probe packets to node $j$.

Note that, in case 2) and 3), the VLAN id of probe packets is reset such that the monitor can distinguish the probe packets receiving from different paths. By adding multiple actions to a rule, monitor node $s$ sends probe packets to the other nodes in a multicast manner, i.e., only one probe packet is transmitted on each link of the probing tree in each round of probing. In contrast, the probe packets returned from non-monitor node (carried on reverse paths) are forwarded in a unicast manner, i.e., each reverse path will carry a different probe packet in each round of probing.

The probe packets sent back to monitor $s$ are forwarded along the reverse paths on the probing tree, and the destination IP address of these probe packets is $ip_s$. So we install only one rule with "matching fields" $(*, ip_s)$ in each node to match the returning probe packets, and the action of the rule in node $i$ is to just simply forward the matched probe packets to the parent node of node $i$.

TABLE I

THE FLOW RULES DESIGNED FOR PROBING PACKETS

| NO. | Match field | Actions |
|---|---|---|
| 1 | $(*, ip_d)$ | **for** each adjacent node $j$<br>**if** node $j$ is a child of node $i$ on $T(V_T, L_T)$<br>forward the packet $pkt$ to node $j$<br>**end if**<br>$pkt.dip = ip_s$<br>$pkt.vlanid = I_{ij}$<br>**if** node $j$ is a parent of node $i$ on $T(V_T, L_T)$<br>forward the packet $pkt$ to node $j$<br>**end if**<br>**if** link $(i, j)$ is a non-tree link<br>forward the packet $pkt$ to node $j$<br>**end if**<br>**end for** |
| 2 | $(*, ip_s)$ | forward packets to the parent of node $i$. |

TABLE II

THE RULES INSTALLED IN SDN NODE $i$ OF A HYBRID SDN NETWORK

| NO. | Match field | Actions |
|---|---|---|
| 1 | $(ip_s, *)$ | **for** each measurement path $p_{ij}$<br>**if** $p_{ij} \in P_s$<br>$pkt.dip = ip_j$<br>**if** node $j$ is a child of node $i$ on $T_a(V_a^T, L_a^T)$<br>Forward the packet $pkt$ to node $k_{ij}$<br>**end if**<br>**if** node $j$ is a parent of node $i$ on $T_a(V_a^T, L_a^T)$<br>$pkt.sip = ip_d$<br>$pkt.vlanid = I_{ij}$<br>Forward the packet $pkt$ to node $k_{ij}$<br>**end if**<br>**if** link $(i, j) \notin L_a^T$<br>Forward the packet $pkt$ to node $k_{ij}$<br>**end if**<br>**end if**<br>**if** $p_{ij} \in P_h$<br>$pkt.sip = ip_d$<br>$pkt.dip = ip_i$<br>Forward the packet $pkt$ to node $j$<br>**end if**<br>**if** $p_{ij} \in P_r$<br>$pkt.dip = ip_j$<br>Forward the packet $pkt$ to node $k_{ij}$<br>**end if** |
| 2 | $(ip_d, *)$ | Forward packets to $k_{ij}$ ($j$ is the parent node of node $i$ on $T_a(V_a^T, L_a^T)$) |

In summary, for a full SDN network, only two rules are required in each node to realize the constructed measurement paths: one is used to forward probe packets in the forward direction, and another is used to forward probe packets in the reverse direction. The two flow rules installed in node $i$ are listed in table II.

### C. Monitor Placement

For each round of probing, a probe packet is multicasted to all nodes along the probing tree from the monitor placed at node $s$, two probe packets are transmitted on each non-tree link (on two directions), and the returning probe packets are sent back to the monitor along reverse paths, i.e., the total number of probe packets transmitted on forward paths (probing tree) and non-tree links is $2m-n+1$, and the total number of probe packets transmitted on reverse paths is $\sum_{i \in V} (\gamma_i + 1) \cdot len_i^s$, where $\gamma_i$ and $len_i^s$ denote the number of non-tree links incident

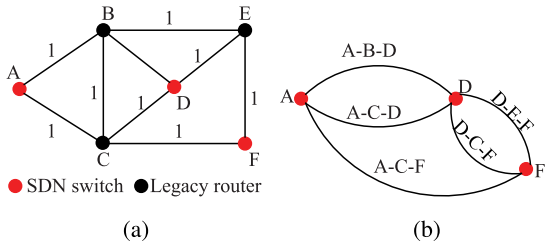



Fig. 2. An illustrative example for constructing overlay network. (a) A hybrid SDN network. (b) The overlay network.

to node $i$ and the length the reverse path $p_i^r$ on the probing tree root at node $s$, respectively. Thus, the total number of probe packets injected to the network in each round of probing is $2m - n + 1 + \sum_{i \in V} (\gamma_i^s + 1) \cdot len_i^s$. To reduce the length of measurement paths, ProLIMI uses the shortest-path tree as the probing tree. For simplicity, we define the probing cost of node $s$ as $\sum_{i \in V} (\gamma_i^s + 1) \cdot len_i^s$. Evidently, to minimize the number of probe packet transmitted in a full SDN network, the monitor should be placed to the node $s$ with minimum probing cost.

## V. RTLM Identification in Hybrid SDN Networks

Due to cost constraint, incremental deployment of SDN (i.e., hybrid SDN) becomes a natural choice of network operators. In this section, we will present how to identify RTLMs in hybrid SDN networks. We first show that the monitor placement problem in hybrid networks is NP-hard. Then we formulate the monitor placement and measurement path selection problem, and to solve the problem efficiently, we propose a heuristic algorithm. Lastly, we discuss how to design flow rules installed on SDN switches to realize measurement paths.

### A. The Monitor Placement and Measurement Path Selection Problem in Hybrid SDN Networks

*Theorem 1:* The monitor placement problem for hybrid SDN networks is NP-hard.

*Proof:* See the technical report [37]. □

Due to the routing constraint, some paths in hybrid SDN networks may be infeasible (cannot be realized). For example, the path A - B - D - E in Fig. 2 (a) is infeasible since node $D$ is not on the shortest path from node $B$ to node $E$. A feasible path must satisfy the shortest path routing constraint and be loop-free. Specifically, a feasible path in hybrid SDN networks is defined as follows.

*Definition 1 (Feasible Path):* The path $p_{st}$ from node $s$ to node $t$ in a hybrid SDN network $G(V, L)$ is a feasible path if it meets the following two constraints:

**c1)** For each link $(u, v) \in p_{st}$, $|sp_{ut}| = w_{uv} + |sp_{vt}|$ and $|sp_{vs}| = w_{uv} + |sp_{us}|$ if $u$ and $v$ are legacy IP routers.

**c2)** For each link $(u, v) \in p_{st}$, $|sp_{vt}| \leq w_{uv} + |sp_{ut}|$ and $|sp_{vs}| = w_{uv} + |sp_{us}|$ if $u$ is an SDN switch and $v$ is a legacy IP router.

**c3)** For each link $(u, v) \in p_{st}$, $|sp_{us}| \leq w_{uv} + |sp_{vs}$ and $|sp_{ut}| = w_{uv} + |sp_{vt}|$ if $u$ is a legacy IP router and $v$ is an SDN switch.

For **Definition 1**, the equations in constraints **c1)**, **c2)** and **c3)** ensure that the shortest path routing constraint is satisfied, and the inequations in constraints **c2)** and **c3)** guarantee that the path is loop-free (e.g., if $|sp_{vt}| > w_{uv} + |sp_{ut}|$ in **c2)**, a packet sent from SDN switch $u$ to destination $t$ will be forwarded back to SDN switch $u$ by legacy IP router $v$, thus resulting in a routing loop between nodes $u$ and $v$.).

The feasible paths, which are used to provide round-trip measurements in ProgLIMI, are called feasible measurement path. Base on the definition of feasible path, the feasible measurement path can be defined as follows.

*Definition 2 (Feasible Measurement Path):* The path $p_{st}$ from node $s$ to node $t$ in a hybrid SDN network $G(V, L)$ is a feasible measurement path if it meets the following two constraints:

**c4)** The path $p_{st}$ is a feasible path.

**c5)** For $\forall$ node $i \in p_{st}$, $i \neq s$, and $i \neq t$, node $i$ must be a legacy router.

Among all feasible paths, we only consider the feasible measurement paths, which are feasible paths and satisfy the additional constraint **c5)**. This is because the metrics of other feasible paths can be calculated according to the metrics of feasible measurement paths. Let $P$ denote the set of feasible measurement paths. For ease of description, we classify feasible measurement paths into three types, which are denoted by $P_s$, $P_h$ and $P_r$ ($P = P_s \cup P_h \cup P_r$), respectively. In the following, we will introduce the three types of feasible measurement paths used by ProgLIMI.

$P_s$ is the set of feasible measurement paths between two SDN switches. To identify the round-trip metrics of feasible measurement paths in $P_s$, we construct an overlay network $G_o(V_o, L_o)$ for $G(V, L)$. For each feasible measurement path $p_{uv} \in P_s$ between SDN switches $u$ and $v$, a link $(u, v)$ is added in $G_o(V_o, L_o)$. Fig. 2(b) shows the overlay network for the hybrid SDN network in Fig. 2(a). Obviously, the constructed overlay network can be seen as a full SDN network, and the RTLMs of an overlay network can be identified using the method introduced in Section IV, with only one monitor placed at an SDN switch. Namely, all the round-trip metrics of feasible measurement paths in $P_s$ can be easily measured. To reduce the probing cost, a monitor is placed at the SDN node with the minimum probing cost in the overlay network (see IV. C).

With the monitor placed in $G_o(V_o, L_o)$, we also can easily identify the round-trip metric of a feasible measurement path between an SDN switch $u$ and an IP router $v$ if the feasible measurement path includes only one link as follows. When the SDN switch $u$ receives a probe packet, it first makes a copy and modifies the destination IP address of the probe packet to its IP address $ip_u$. Upon node $v$ receiving the probe packet, it will send the probe packet back to SDN node $u$, which will send it to the monitor. Thus the monitor can obtain the round-trip metric of the one-hop feasible measurement path by subtracting the round-trip metric of the feasible measurement path between node $u$ and the monitor node from the round-trip metric of the feasible measurement path between node $v$ and the monitor. Notably, if the number of hops of a path between an SDN switch $u$ and IP router $v$ is greater than 1, the round-

trip metric of the path cannot be measured by using the above method. In this case, the probe packets with destination IP address $ip_u$ will be forwarded back to node $u$ at the first hop of the path. Thus, the round-trip metric of the path cannot be measured. We use $P_h$ to denote the set of above feasible measurement paths whose round-trip metrics can be measured using the monitor placed at an SDN switch.

However, only using the measurements from the aforementioned feasible measurement paths may not be sufficient for identifying all the RTLMs. So in such case, more monitors will need to be placed at IP routers such that $m$ linearly independent feasible measurement paths can be established between monitors. Let $P_r$ denote the set of feasible measurement paths which are not in sets $P_s$ and $P_h$, i.e., $P_r = P \backslash (P_s \cup P_h)$. If a feasible measurement path $p \in P_r$ is selected as a measurement path, the endpoints of path $p$ must be equipped with monitors if they are not SDN switches. So we only need to select measurement paths and place monitors at the IP routers which are the endpoints of the selected measurement paths in $P_r$.

Given the all pair shortest path distances, the feasible measurement paths can be easily calculated using Breadth First Searching (BFS) algorithm. For simplicity, we do not consider Equal-Cost Multi-Paths (ECMP) between two IP legacy routers when calculating feasible measurement paths, i.e., we only choose one of the shortest paths between two IP legacy routers. Given the feasible measurement paths, we formulate the monitor placement and measurement path selection problem in hybrid SDN networks as a Mixed Integer Linear Programming (MILP) in [37]. However, solving the MILP is computationally expensive and even infeasible in large networks. Hence, to efficiently solve the RTLM identification problem in large hybrid SDN networks, we propose a heuristic algorithm in the next section.

### *B. The Monitor Placement and Measurement Path Selection Algorithm*

To uniquely identify the RTLMs of a hybrid SDN network, we need to place some monitors in the network and select $m$ linearly independent feasible measurement paths. In this section, we propose an efficient heuristic algorithm called Monitor Placement and Measurement Path Selection (MP-MPS) [5] to place monitors and select feasible measurement paths for hybrid SDN networks. For the convenience of description, we introduce the following definition.

*Definition 3 (Identifiable and Unidentifiable Link):* Given a set of linearly independent paths, if the round-trip metric of a link can be uniquely inferred by measuring the set of paths, the link is identifiable, otherwise, the link is unidentifiable.

MP-MPS places monitors and selects measurement paths simultaneously. **Algorithm 2** shows the detailed procedure of MP-MPS. MP-MPS first constructs an overlay network $G_o(V_o, L_o)$ for the hybrid SDN network $G(V, L)$. The constructed overlay network $G_o(V_o, L_o)$ is a full SDN network. Then to identify RTLMs of $G_o(V_o, L_o)$, a monitor is placed to the SDN node $s$ that has the minimum overlay probing cost (the probing cost of node $s$ in $G_o(V_o, L_o)$). By using the method for full SDN networks, all RTLMs of the overlay network can be identified. $P_s$ is the set of feasible measurement paths represented by the links of $G_o(V_o, L_o)$. To select linearly independent paths from set $P_s$ (line 4), we use the **Algorithm 3**, which is a variant of $QR$ decomposition with column pivoting [24], [25]. In **Algorithm 3**, $\|\cdot\|_2^2$ denotes square of the 2-norm of a vector. **Algorithm 3** incrementally decomposes the routing matrix $A$ into $QR$ ($Q \in \mathbb{R}^{m \times h}$, $R \in \mathbb{R}^{h \times h}$), where $Q$ is a matrix with orthonormal columns, $R$ is an upper triangular matrix and $h$ is the number of linearly independent paths that have been selected. However, it is generally impossible to uniquely identify all RTLMs by only using the feasible measurement paths between SDN switches, i.e., the number of linearly independent shortest paths in set

**Algorithm 2** Monitor Placement and Measurement Path Selection (MP-MPS)

**Input:** A hybrid SDN network $G(V, L)$, and the feasible measurement path sets $P_s$ and $P_h$.

**Output:** A subset $M$ of nodes in $V$ selected as monitor nodes, and a set $MP$ of $m$ linearly independent feasible measurement paths.

1: Construct an overlay network $G_o(V_o, L_o)$ for $G(V, L)$ based on the paths in $P_s$
2: Place a monitor to the SDN node $i$ with the minimum overlay probing cost, and node $i$ is inserted into set $M$
3: $A \leftarrow NULL$, $R \leftarrow NULL$
4: $(MP, A, R) \leftarrow$ **Algorithm 3** $(P_s, A, R)$ (Select linearly independent paths from set $P_s$)
5: **if** $|MP| == m$ **then**
6: **return** $(M, MP)$
7: **end if**
8: $(MP, A, R) \leftarrow$ **Algorithm 3** $(P_h, A, R)$ (Select linearly independent paths from set $P_h$)
9: **if** $|MP| == m$ **then**
10: **return** $(M, MP)$
11: **end if**
12: **for** each link $(u, v) \in L$ **do**
13: Let path $p \leftarrow \{u, v\}$
14: $R_{12} \leftarrow {R^{-1}}^T A a_p^T = Q^T a_p^T$
15: **if** $R_{12} \neq 0$ **then**
16: Push nodes $u$ and $v$ to stack $S$
17: **end if**
18: **while** $S$ is not empty **do**
19: $z \leftarrow pop(S)$
20: **if** $z \notin M$ **then**
21: Select $z$ as a monitor and insert $z$ into $M$
22: Let $SP \leftarrow \emptyset$ and add the feasible measurement paths from node $u$ to the nodes in $M \cup V_{SDN}$ to set $SP$
23: $(TP, A, R) \leftarrow$ **Algorithm 3** $(SP, A, R)$
24: $MP \leftarrow MP \bigcup TP$
25: **if** $|MP| == m$ **then**
26: **return** $(M, MP)$
27: **end if**
28: **end if**
29: **end while**
30: **end for**

**Algorithm 3** Linearly Independent Path Selection (LIPS)

**Input:** A set $CP$ of candidate paths, and matrices $A$ and $R$
**Output:** A set $LP$ of linearly independent paths in $CP$, and the updated matrices $A$ and $R$

1: **for** each path $p \in CP$ **do**
2: **if** $A == NULL$ **then**
3: $A \leftarrow [a_p]$ and $R \leftarrow \left[\|a_p\|_2^2\right]$
4: **else**
5: $R_{12} \leftarrow R^{-T} A a_p^T = Q^T a_p^T$
6: $R_{22} \leftarrow \|a_p^T\|^2 - \|R_{12}\|_2^2$
7: **if** $R_{12} \neq 0$ **then**
8: Add path $p$ to set $LP$
9: Update $R \leftarrow \begin{bmatrix} R & R_{12} \\ 0 & R_{22} \end{bmatrix}$ and $A \leftarrow \begin{bmatrix} A \\ a_p \end{bmatrix}$
10: **end if**
11: **end if**
12: **end for**
13: **return** $(LP, A, R)$

$MP$ is less than $m$. Therefore, MP-MPS will select more IP routers to place monitors in the following steps. First, MP-MPS selects linearly independent paths from feasible measurement path set $P_h$ and adds these paths to linearly independent measurement path set $MP$ (line 8). Then, for an unidentifiable link $(u, v)$ (the path only traversing link $(u, v)$ is linearly independent with the paths in set $MP$), MP-MPS will sequentially select nodes $u$ and $v$ as monitors (lines 12-21) if nodes $u$ and $v$ are IP routers, and the feasible measurement paths, which are starting from $u$ or $v$ to the SDN nodes and monitor nodes in set $M$, will be added to set $MP$ if they are linearly independent with selected paths in $MP$ (lines 23-24). MP-MPS terminates when the number of selected linearly independent paths equals $m$ (lines 6, 10 and 26). **Algorithm 2** has the time complexity of $O(K \cdot m^2)$ [37], where $K$ is the total number of feasible measurement paths in the network.

### C. Flow Rule Design

By using **Algorithm** 2, we can obtain a set of linearly independent measurement paths and a set of monitor nodes. To measure the round-trip metrics of the measurement paths with the minimum cost, we need to carefully design the flow rules for probing flows. Let $p_{ij}$ denote a measurement path from node $i$ to node $j$. If nodes $i$ and $j$ are IP routers, path $p_{ij}$ is a shortest path (see **Definition 2**) and nodes $i$ and $j$ are monitor nodes, according to **Algorithm** 2. To measure the round-trip metric of path $p_{ij}$, monitor node $i$ sends a probe to monitor node $j$, and monitor node $j$ returns the received probe to monitor $i$. In this case, the probe packets are forwarded by IP routers along the shortest path $p_{ij}$, and thus we do not need to design rules for realizing $p_{ij}$.

If both node $i$ and node $j$ are SDN switches, we need to use rules installed in SDN switch to forward probe packets correctly. To facilitate the flow rule design process, we construct an auxiliary network $G_a(V_a, L_a)$ as follows. For each measurement path $p_{ij}$, add a link $(i, j)$ to $G_a(V_a, L_a)$ if both node $i$ and node $j$ are SDN switches. Fig. 3(a) shows the auxiliary network $G_a(V_a, L_a)$ for the network in Fig. 2(a).

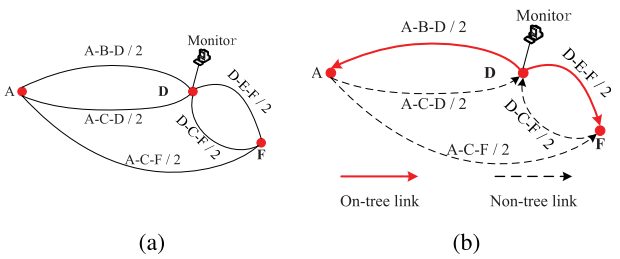



Fig. 3. An auxiliary network and its probing tree. (a) The auxiliary network. (b) The probing tree.

$G_a(V_a, L_a)$ is a full SDN network. Therefore, we can use the method for full SDN networks to measure RTLMs and design flow rules for the SDN switches of $G_a(V_a, L_a)$. Let $s$ be the SDN switch placed with monitor and $T_a(V_a^T, L_a^T)$ be the probing tree rooted at $s$ for $G_a(V_a, L_a)$. For the example in Fig. 3(a), the monitor is placed at node $D$, and the probing tree is shown in Fig. 3(b) using red lines. Since a link $(i, j)$ in $G_a(V_a, L_a)$ represents a measurement path $p_{ij}$, when a node $i$ forwards a probe packet to node $j$ in $G_a(V_a, L_a)$, it has to first forward the probe packet to its adjacent node $k_{ij} \in p_{ij}$. And to guarantee that the probe packet can be forwarded to node $j$ along path $p_{ij}$, node $i$ needs to modify the destination IP address of the probe packet to $ip_j$ (the IP address of node $j$).

At last, if either node $i$ or node $j$ is an SDN switch, we should consider the following two cases: 1) the measurement path $p_{ij}$ includes only one link $(i, j)$, i.e., $p_{ij} \in P_h$; 2) the measurement path $p_{ij}$ includes two or more links. Without loss of generality, we assume that node $i$ is an SDN switch and node $j$ is a legacy IP router. For the first case, SDN node $i$ should copy a probe packet and modify the destination IP address of the probe packet to $ip_i$, and forwards the probe packet to node $j$. For the second case, SDN node $i$ should copy a probe packet and modify the destination of the probe packet to $ip_j$, and forwards the probe packet to its adjacent node $k_{ij} \in p_{ij}$. Based on the above description, we can get the two rules (see Table II) installed on an SDN switch $i$ for forwarding probe packets in a hybrid SDN network.

## VI. Performance Evaluation

To evaluate the feasibility and performance of ProgLIMI, we build a physical testbed and conduct simulations in Mininet [26]. In both testbed and simulation evaluations, we consider two different link metrics: delay and loss rate.

The round-trip delay of a link $(u, v)$ consists of propagation delay $d_{uv}^p$ and queuing delay $d_{uv}^q$, where $d_{uv}^p$ is fixed and $d_{uv}^q \sim exp(\lambda)$. The round-trip propagation delay of every link is set to 6*ms*, and the mean $\lambda$ for each link is randomly chosen in [5, 10]*ms*. The round-trip loss rate of each link is generated uniformly in the interval [1%, 5%]. Delay and packet loss are introduced using NetEm [27]. In operational networks, link metrics may fluctuate frequently over time. However, the frequent fluctuation does not reflect significant and persistent changes in network performance or trends [1], [28]. The goal of the link metric identification system is to capture and locate significant change of link metrics, which are usually used for troubleshooting. The study in [29] and [30] show that many significant network delay fluctuations are caused

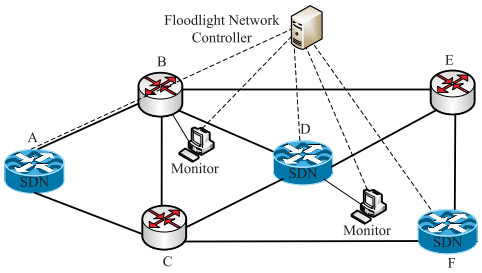



Fig. 4. The hybrid SDN network testbed topology.

by routing changes, and the routing changes usually take place on a timescale of tens of minutes. Therefore, we assume that link metrics vary every 10 minutes, and the measurement time windows are 60 minutes. However, ProgLIMI can also be used in the scenarios, where the link metrics significantly change in a short time period. So in a few simulations, the link metric change interval and the measurement time window are set to 10 and 60 seconds, respectively. The monitors respectively send probe packets to the network with time intervals 100*ms* and 2*ms* where the link metric change intervals are set to 10 minutes and 10 seconds, respectively. For convenience, $t_p$, $t_c$, and $t_w$ denote the probing packets sending interval, the link metric change interval, and the measurement time window, respectively. Similar to [17] and [20], the only background traffic in the network is OpenFlow communication between the controller and the switches.

We use Mean Relative Error (MRE) as the performance metric to evaluate the RTLM identification accuracy of ProgLIMI. Let $x_i$ and $\hat{x}_i$ denote the real RTLM and identified RTLM in the $i$th measurement, respectively. $N$ is the number of measurements carried out under each setting. The MRE is defined as:

$$MRE = \frac{1}{N}\sum_{i=1}^{N}\frac{|x_i - \hat{x}_i|}{x_i} \times 100\% \tag{3}$$

### A. Testbed Experiments

We build a hybrid SDN network testbed as shown in Fig. 4. Here, we do not consider the full SDN scenario since full SDN network is just a special case of the hybrid network. The testbed consists of three SDN switches and three legacy routers. The monitors are placed at node $B$ and $D$, and all link weights are 1. The SDN switches and legacy routers are implemented on Intel Xeon Quad Core servers with 8 GB memory and 1 Gbps NICs using Open vSwitch (OVS). The flow rules of legacy routers are preconfigured such that the probe packets can be forwarded along the shortest paths at these routers. The SDN switches are controlled by a floodlight network controller, and ProgLIMI runs on the network controller.

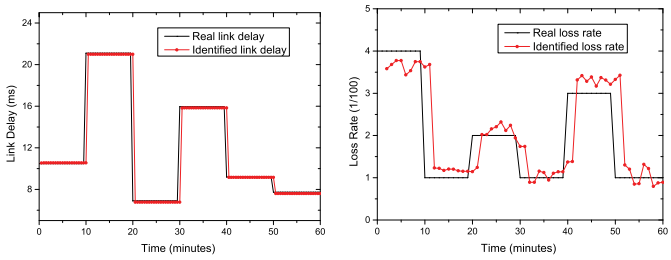
Table III shows the MREs of the identified Round-Trip Link Delays (RTLDs) of the testbed network. As shown in Table III, the MREs of the identified RTLDs are very low. Through analyzing the log data in our experiments, we found that the RTLD identification errors are mainly caused by the overhead of executing actions in flow rules of SDN switches. We measured the overhead in our testbed

TABLE III
MREs OF THE IDENTIFIED RTLDs OF THE TESTBED NETWORK

| Link | (A, B) | (A, C) | (B, D) | (B, C) | (B, E) | (D, C) | (D, E) | (C, F) | (F, E) |
|---|---|---|---|---|---|---|---|---|---|
| MRE (‰) | 3.3 | 3.6 | 5.6 | 1.7 | 1.3 | 2.1 | 2.3 | 1.9 | 3.2 |

TABLE IV
MREs OF THE IDENTIFIED RTLLRs OF THE TESTBED NETWORK

| Link | (A, B) | (A, C) | (B, D) | (B, C) | (B, E) | (D, C) | (D, E) | (C, F) | (F, E) |
|---|---|---|---|---|---|---|---|---|---|
| MRE (%) | 4.1 | 2.8 | 1.8 | 10.5 | 5.6 | 10.2 | 8.7 | 2.3 | 9.6 |




(a) (b)

Fig. 5. The real and identified round-trip delay and round-trip loss rate of link $(E, F)$ vary over time. (a) Link delay. (b) Link loss rate.

network. The measured results reveal that the average latency for modifying an IP address or a VLAN tag is about 12*us*, and the average latency for duplicating and forwarding a packet is about 9*us*. In Table III, we also can observe that different links have different MREs. This is because the probe packets forwarded on different measurement paths will be processed by different sets of actions. Thus, the extra latencies incurred are different.

We also evaluated the Round-Trip Link Loss Rate (RTLLR) identification performance of ProgLIMI on the testbed network, and the results are shown in Table IV. The MREs of the identified RTLLRs are also low. For example, the highest MRE of the identified RTLLR is less than 11%. However, the MREs of the identified RTLLRs are much higher than the MREs of the identified RTLDs. The reason is that in our experiments, the RTLLRs are low (range from 1% to 5%), and thus a small deviation may lead to large measurement error. To mitigate the problem, we can increase the probe packet sending frequency. But it will incur higher overhead to the network.

In order to evaluate the reaction of ProgLIMI when RTLMs vary over time, we plot the identified as well as real round-trip delay and round-trip loss rate of link $(E, F)$ over time in Fig. 5. From Fig. 5(a), we can see that the two curves overlap with each other. It demonstrates that the RTLM identification method can capture the RTLD variation in a very short time. In contrast, the identified RTLLR slightly lags behind the real RTLLR (Fig. 5(b)). This is because computing the RTLLR requires counting the received probe packets in a time period, which inevitably yields time delay to compute RTLLR. However, this is not a crucial issue for the practical networks since the status of practical networks always varies in the time scale of minutes or even hours [32], which is much longer than the time required for completing a measurement.

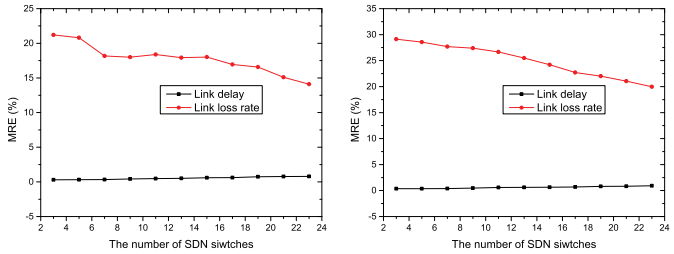



Fig. 6. The MREs of identified RTLD and RTLLR under different number of SDN switches in Mininet simulations. (a) $t_c = 10m$, $t_w = 60m$. (b) $t_c = 10s$, $t_w = 60s$.

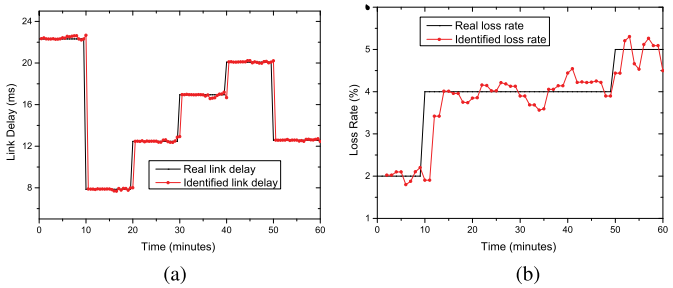



Fig. 7. The real and identified delay and loss rate of link $(16, 19)$ vary over time when the link delay and loss rate are constant in each period. (a) Link delay. (b) Link loss.

As presented in subsections IV. B and V. D, ProgLIMI uses two flow rules in each SDN switch to forward probe packets. The two flow rules have multiple actions (Table I and II), which are defined in OpenFlow specification [33]. To check whether the flow rules with multiple actions are supported by commodity switches, we make a survey on three SDN switches from different vendors: Pica8 P-3297, Centec V580, and NoviFlow NoviKit 250. We found that all the three SDN switches support the rules with multiple actions used in ProgLIMI [34]–[36]. To further verify that ProgLIMI is realizable, we conduct an experiment by replacing OVSs with Pica8 P-3297 SDN switches in our testbed. The experiment verifies that ProgLIMI also works well. In addition, we also can use the group table with multiple action buckets to realize the same function of the two flow rules designed by ProgLIMI. The group table is widely supported by commodity switches.

### *B. Mininet Simulations*

To evaluate the performance of ProgLIMI in larger network topologies, we develop a simulation platform with Mininet [26]. We conduct simulations on GEANT topology [37, Fig. 8] with the different number of SDN switches. For a given number of SDN switches, the nodes with the higher degree have higher priority to deploy SDN switches, and if there is a tie, the nodes are ordered arbitrarily.

*1) MRE:* Fig. 6(a) plots the average MREs of identified RTLDs and RTLLRs for all links under different number of SDN switches when the RTLM change interval is set to 10 minutes. Generally, the MREs of identified RTLDs and RTLLRs in simulations is higher than that in the testbed experiments. The reason can be explained as follows: 1) The MRE of RTLD is mainly caused by the latency of executing actions in a flow rule, and latencies for executing the actions in Mininet is higher than that in our physical testbed (in Mininet, modifying an IP address or a VLAN tag takes about $17us$, and duplicating and forwarding a packet take about $13us$.). 2) The MRE of RTLLR is mainly affected by the length of measurement paths, i.e., the MRE of RTLLR increases with the lengths of measurement paths (measurement error is accumulated at each hop). Since the topology used in the simulation is larger than the testbed network topology, the lengths of the measurement paths are increased accordingly. We can see that similar to the results obtained from the testbed, the MREs of identified RTLDs are also very low, e.g., the maximum average MRE of identified RTLD is 8‰. As discussed above, the MRE of identified RTLD is mainly affected by the extra latencies for executing actions. To quantify the extra latencies, we measured the extra time taken for executing actions on the measurement paths of a probe packet by setting all link delays to 0. The average extra time is $0.28ms$, which is negligible since the average round-trip delay of measurement paths is at least tens of milliseconds. This explains why the MREs of identified RTLDs are very low. At last, we can observe that the MREs of identified RTLLRs are relatively high, e.g., the maximum average MRE of identified RTLLR is 21.2%. If we want to get the precise RTLLR, the MRE of identified RTLLR may be high. However, in most cases, we only need to capture the significant change of RTLLR (e.g., used for troubleshooting), which is not very sensitive to the MRE.

To evaluate the MREs of identified RTLMs under the scenario where the link metrics significantly change in a short time period, we conduct similar simulations in Mininet by setting the link metric change interval and probe packets sending interval to $10s$ and $2ms$, respectively. The simulation results are shown in Fig. 6(b). The curves in Fig. 6(b) follow the similar trend as in Fig. 6(a). From Fig. 6(b), we can see that the MREs of identified RTLDs is also very low, and the MREs of identified RTLLRs are slightly higher than that in Fig. 6(a). In addition, we also found that the MRE can be improved by reducing the probe packets sending interval. The results verify that ProgLIMI can also be used in the scenarios, where the link metrics change significantly in a short time period.

Moreover, Fig. 7 shows the identified and real round-trip delay and loss rate of the link (16, 19) over time when the number of SDN switches is set to 7 (SDN switches are placed at nodes 1, 2, 6, 9, 12, 15, and 16). Similar to the results obtained on the testbed, ProgLIMI can capture the RTLD variations and has a slight lag for identifying the RTLLR. However, the overall performance is satisfactory for most network management applications.

In Fig. 7(a), the link delays are assumed to be constant in each period. However, the link delays of real networks may slightly fluctuate over time. So to evaluate how ProgLIMI performs under such cases, we carry out a simulation where the link delays are assumed to be random variables following the normal distribution with mean $\theta$ and standard deviation $\sigma$. The mean $\theta$ changes for every 10 minutes, and the standard deviation $\sigma$ is set to 0.5. We reset the link delays randomly every 5 seconds. Fig. 8 shows the average MREs of identified RTLDs for each time interval (the length of each time interval is 10 minutes) when $t_p = 10ms$ and $t_p = 20ms$. From Fig. 8,

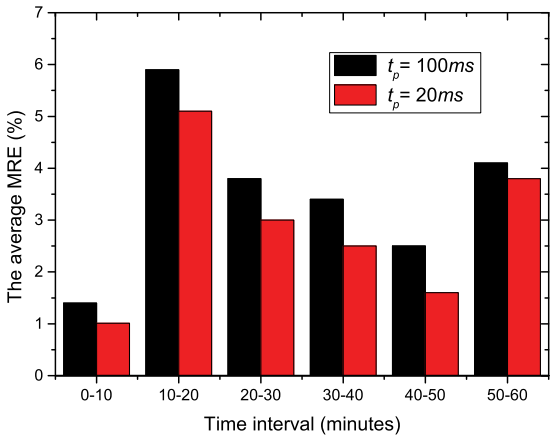



Fig. 8. The average MREs of identified RTLD for each time interval when $t_p = 100ms$ and $t_p = 20ms$.

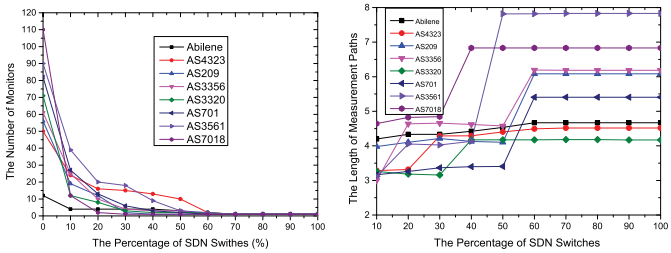
we can see that the average MREs under all cases are also low (less than 6%), and ProgLIMI can also identify the RTLDs with high accuracy even when the probe packets sending frequency is low (10 probe packets per second). In addition, we can observe that higher probing frequency yields lower MREs of identified RTLDs.

ProgLIMI probes the measurement paths by using the active probing method, which will incur extra overhead to a network, including monitors and probe packets transmitted in the network. We evaluate the overhead incurred by ProgLIMI on both synthetic topologies with 100-nodes and real topologies obtained from the Rocketfuel project [31]. We use ERdős-Rényi (ER) and Barabási-Albert (BA) models to generate synthetic topologies. In ERdős-Rényi model, a topology is constructed by independently connecting each pair of nodes by a link with a fixed probability $p$. BA model generates random topology by beginning with an initially connected topology of $n_0$ ($n_0 = 4$ in our simulations) nodes and adding new nodes sequentially. Each new node is connected to $d_{min}$ existing nodes with a probability that is proportional to the degree of the existing nodes. Generally, for a given number of monitors and a random topology, ProgLIMI cannot guarantee to achieve full RTLM identifiability. Therefore, we evaluate its measurement cost by using the average number monitors required to achieve full RTLM identifiability under the different number of SDN switches ($k$). As expected, the number of monitors required by ProgLIMI to identify all RTLMs decreases dramatically with the increase of $k$ in synthetic topologies. And when $k \geq 50$, ProgLIMI can achieve full RTLM identifiability with only one monitor in all the synthetic topologies. However, due to space limitations, we omit the simulation results for synthetic topologies here and refer interested readers to [37] for details.

We also conduct a set of simulations on 8 ISP topologies derived from the Rocketfuel project [31]. Table V summarizes the number of nodes and links in each topology. The number of monitors required by ProgLIMI for the 8 ISP topologies under different percentages of SDN switches is shown in Fig. 9(a). As shown in Fig. 9(a), the number of monitors required by ProgLIMI also decreases rapidly with the increasing number of SDN switches. Comparing with the results in synthetic

TABLE V
REAL ISP TOPOLOGIES

| Topology | Abilene | AS4323 | AS209 | AS3356 | AS3320 | AS701 | AS3561 | AS7018 |
|---|---|---|---|---|---|---|---|---|
| $\|V\|$ | 12 | 51 | 58 | 63 | 70 | 83 | 92 | 115 |
| $\|L\|$ | 37 | 161 | 108 | 285 | 355 | 219 | 329 | 148 |




(a) (b)

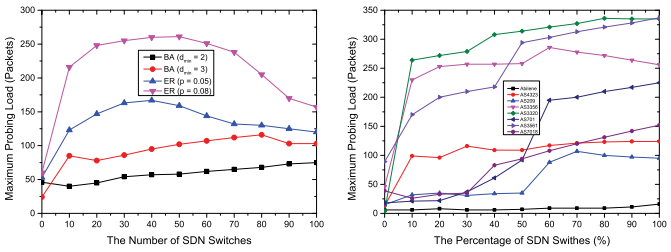
Fig. 9. The number of monitors required and the average length of measurement paths in real ISP topologies under different percentages of SDN switches. (a) The number of monitors. (b) The length of measurement paths.

topologies, ProLIMI needs the higher fraction of nodes to be monitors in real topologies when the number of SDN switches is small (less than 30%). One reason is that in these real topologies, a large number of nodes have degree less than 3, which have to be selected as monitors if their neighbors are not SDN switches. It is verified in [13] that in some real topologies, more than 60% of nodes need to be monitor even if the explicit routing is allowed. Based on the results in synthetic and real topologies, we can conclude that the number of monitors required in SDN networks to achieve full RTLM identifiability is far less than that in legacy IP networks.

Fig. 9(b) shows the average length of the measurement paths (in hops) in the ISP topologies. Generally, the length of the measurement paths increases by increasing the number of SDN switches. This is because the number of required monitors in a network decreases with the increasing number of SDN switches (as shown in Fig. 9(a)). In the extreme case (i.e., where all nodes are SDN-enabled), all measurement paths must start and end at a single monitor node, which leads to longer measurement paths. And in all the topologies, the average length of measurement paths is less than 8 hops.

We assume that for each round of probing, a monitor node sends or receives a probe packet, which is forwarded along the measurement paths. Since multiple measurement paths may go through the same link, a link may carry multiple probe packets in each round of probing. Let $\theta_l$ denote the number of probe packets transmitted on link $l$ in a round of probing. To evaluate the bandwidth overhead incurred by the probing process, we compute the maximum probing load $\theta_{max}$, which is defined as $\theta_{max} = \max\{\theta_l, l \in L\}$.

Fig. 10(a) shows the maximum probing loads of synthetic topologies. Interestingly, the maximum probing loads for BA and ER topologies follow different trends. For ER topologies, the maximum probing loads increases with the number of SDN switches, while for BA topologies, the maximum probing loads first increase and then decrease with the number of SDN switches. This is mainly determined by the topology characteristics. As expected, the maximum probing loads of densely connected topologies are higher than that of sparsely




(a) (b)

Fig. 10. The maximum probing loads of synthetic and real topologies under different percentages of SDN switches. (a) Synthetic topologies. (b) Real topologies.

connected topologies. The reason is that more measurement paths are needed to achieve full RTLM identifiability, and thus more measurement paths may go through the same link. For BA topologies, the degrees of a few nodes are very high, and the degrees of the rest nodes are low. And in our simulations, the SDN switches will be deployed at the nodes with high degree. It implies that the number of hops of measurement paths in BA topologies is much smaller than that in ER topologies. This is why the maximum probing load of BA topologies is much lower than that of ER topologies. For the densely connected BA topologies, the maximum probing loads is less than 270 packets/round. Namely, if the size of a probe is 100 Byte and ProgLIMI conducts 10 rounds of probing each second, ProgLIMI will consume about 2 Mbps bandwidth on that link. We can reduce the bandwidth consumption by reducing the probing frequency. This is possible for RTLD probing. However, reducing the probing frequency may increase the probing error for RTLLR if the link loss rate is low.

Fig. 10(b) shows the maximum probing loads of the ISP topologies. Since these ISP topologies have different characteristics, the maximum probing loads of the ISP topologies also exhibit two different trends with the increasing number of SDN switches. Among the ISP topologies, AS3356 and AS3320 have higher maximum probing loads than other topologies in most case. The reason is that AS3356 and AS3320 have more links than other topologies. It implies that to achieve full RTLM identifiability, AS3356 and AS3320 will use more measurement paths, and thus more probe packets may go through the same link incident to monitors. If the size of a probe is 100 Byte and ProgLIMI conducts 10 rounds of probing per second, ProgLIMI will consume several Mbps bandwidth on some links of the ISP topologies in the worst cases.

## VII. Discussion

To reduce the monitor deployment and maintenance cost and enhance the system reliability, ProgLIMI tries to minimize the number of monitors placed in a network. However, minimizing the number of required monitors brings a negative impact on the accuracy of RTLM identification, especially for the RTLLR identification (see Fig. 6). The reason is that less monitors will lead to longer measurement paths, and the accumulated measurement error will increase as the lengths of measurement paths increase. To improve the accuracy of RTLM identification, ProgLIMI can restrict the lengths of the measurement paths by placing more monitors in an SDN network. Specifically, the following extensions can be made in ProLIMI for full and hybrid SDN networks.

### A. Full SDN Network

If the lengths of measurement paths are constrained, multiple monitors may be required to be placed in a full SDN network $G(V, L)$ for achieving full RTLM identifiability. In this case, we can model the monitor placement problem as a facility location problem [39]. In the facility location problem, there are a set $F$ of facilities and a set $C$ of cities. Each facility $f_i \in F$ has an opening cost $\alpha_i$. Furthermore, there is a connection cost $\lambda_{ij}$ between facility $f_i \in F$ and city $c_j \in C$. The objective is to open a subset of the facilities and connect each city to an open facility so that the total cost is minimized. The monitor placement problem can be modeled as a facility problem as follows: 1) A monitor $m_v$ placed at a node $v \in V$ can be viewed as a facility $f_v$ located at node $v$; 2) A link $l$ can be viewed as a city $c_l$ that needs to be connected to a facility; 3) The cost of connecting city $c_l$ to facility $f_v$, denoted by $\lambda_{lv}$, is equal to the cost of measuring link $l$ using monitor $m_v$, and $\lambda_{lv} = \infty$ if the lengths of the measurement paths for link $l$ are longer than the predetermined threshold. Thus, by solving the facility location problem, we can get a solution for the monitor placement problem under the length constraint of measurement paths.

### B. Hybrid SDN Network

For hybrid SDN networks, the **Algorithm 2** can be easily extended to handle the length constraint of measurement paths: First, the same method used for full SDN networks can be used to place monitors in the overlay network $G_o(V_o, L_o)$ for a hybrid SDN network $G(V, L)$ (Line 1 in **Algorithm 2**); Second, **Algorithm 2** must check the length constraint when it selects measurement paths (lines 8 and 22), i.e., the feasible measurement paths in sets $P_h$ and $SP$ cannot be selected as measurement paths if their lengths exceed the predetermined threshold.

## VIII. Conclusion

We proposed ProgLIMI, an infrastructure that identifies RTLM from the round-trip metrics of a set of measurement paths for SDN networks. The goal of ProgLIMI is to identify RTLMs with minimum cost by leveraging the programmable routing control capability of SDN. In this paper, we presented how to solve the problems for both full and hybrid SDN networks. Specifically, we introduced how to place monitors, how to construct linearly independent measurement paths, and how to design flow rules such that the measurement paths can be realized. Finally, we have conducted extensive evaluations on both physical testbed and simulation platform. The results have confirmed that ProgLIMI is feasible and can achieve accurate RTLM identification in SDN networks with low measurement overheads (monitors, flow rules, and bandwidth).

## References

[1] C. Guo *et al.*, "Pingmesh: A large-scale system for data center network latency measurement and analysis," in *Proc. ACM SIGCOMM*, 2015, pp. 139–152.

[2] E. Lawrence, G. Michailidis, V. N. Nair, and B. Xic, "Network tomography: A review and recent developments," *Frontiers Statist.*, vol. 54, pp. 345–364, Jul. 2006.
[3] N. G. Duffield and F. Lo Presti, "Multicast inference of packet delay variance at interior network links," in *Proc. IEEE INFOCOM*, Mar. 2000, pp. 1351–1360.
[4] Y. Xia and D. Tse, "Inference of link delay in communication networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 12, pp. 2235–2248, Dec. 2006.
[5] X. Wang, M. Malboubi, S. Wang, S. Xu, and C.-N. Chuah, "Practical approach to identifying additive link metrics with shortest path routing," in *Proc. IEEE Globecom*, Dec. 2015, pp. 1–6.
[6] M.-F. Shih and A. Hero, "Unicast inference of network link delay distributions from edge measurements," in *Proc. IEEE ICASSP*, May 2001, pp. 3421–3424.
[7] F. L. Presti, N. G. Duffield, J. Horowitz, and D. Towsley, "Multicast-based inference of network-internal delay distributions," *IEEE/ACM Trans. Netw.*, vol. 10, no. 6, pp. 761–775, Dec. 2002.
[8] A. Adams *et al.*, "The use of end-to-end multicast measurements for characterizing internal network behavior," *IEEE Commun. Mag.*, vol. 38, no. 5, pp. 152–159, May 2000.
[9] W. Xu, E. Mallada, and A. Tang, "Compressive sensing over graphs," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 2087–2095.
[10] O. Gurewitz and M. Sidi, "Estimating one-way delays from cyclic-path delay measurements," in *Proc. IEEE INFOCOM*, Apr. 2001, pp. 1038–1044.
[11] A. Gopalan and S. Ramasubramanian, "On identifying additive link metrics using linearly independent cycles and paths," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 906–916, Jun. 2012.
[12] A. Gopalan and S. Ramasubramanian, "On the maximum number of linearly independent cycles and paths in a network," *IEEE/ACM Trans. Netw.*, vol. 22, no. 5, pp. 1373–1388, Oct. 2014.
[13] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Identifiability of link metrics based on end-to-end path measurements," in *Proc. ACM IMC*, 2013, pp. 391–404.
[14] L. Ma, T. He, K. K. Leung, D. Towsley, and A. Swami, "Efficient identification of additive link metrics via network tomography," in *Proc. IEEE ICDCS*, Jul. 2013, pp. 581–590.
[15] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Monitor placement for maximal identifiability in network tomography," in *Proc. IEEE INFOCOM*, Apr./May 2014, pp. 1447–1455.
[16] J. Sommers, B. Eriksson, and P. Barford, "On the prevalence and characteristics of MPLS deployments in the open Internet," in *Proc. ACM IMC*, 2011, pp. 445–462.
[17] N. L. M. van Adrichem, C. Doerr, and F. A. Kuipers, "OpenNetMon: Network monitoring in OpenFlow software-defined networks," in *Proc. IEEE INFOCOM*, May 2014, pp. 1–8.
[18] K. Phemius and M. Bouet, "Monitoring latency with OpenFlow," in *Proc. IEEE Conf. Netw. Service Manage. (CNSM)*, Oct. 2013, pp. 122–125.
[19] M. Shibuya, A. Tachibana, and T. Hasegawa, "Efficient performance diagnosis in OpenFlow networks based on active measurements," in *Proc. ICN*, 2014, pp. 268–273.
[20] A. Atary and A. Bremler-Barr, "Efficient round-trip time monitoring in OpenFlow networks," in *Proc. IEEE INFOCOM*, Apr. 2016, pp. 1–9.
[21] S. Vissicchio, L. Vanbever, and O. Bonaventure, "Opportunities and research challenges of hybrid software defined networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 2, pp. 70–75, 2014.
[22] Y. Bejerano and R. Rastogi, "Robust monitoring of link delays and faults in IP networks," *IEEE Trans. Netw.*, vol. 14, no. 5, pp. 1092–1103, Oct. 2006.
[23] C. Yu *et al.*, "Software-defined latency monitoring in data center networks," in *Proc. PAM*, 2015, pp. 360–372.
[24] Y. Chen, D. Bindel, H. Song, and R. H. Katz, "An algebraic approach to practical and scalable overlay network monitoring," in *Proc. ACM SIGCOMM*, 2004, pp. 55–66.
[25] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD, USA: The Johns Hopkins Univ. Press, 1989.
[26] N. Handigol, B. Heller, V. Jeyakumar, B. Lantz, and N. McKeown, "Reproducible network experiments using container-based emulation," in *Proc. CoNext*, 2012, pp. 253–264.
[27] S. Hemminger, "Network emulation with NetEm," in *Proc. Linux Conf. Au*, 2005, pp. 18–23.
[28] H. Burch and C. Chase, "Monitoring link delays with one measurement host," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 33, no. 3, pp. 10–17, 2005.
[29] H. Pucha, Y. Zhang, Z. M. Mao, and Y. C. Hu, "Understanding network delay changes caused by routing events," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 35, no. 1, pp. 73–84, 2007.
[30] I. Cunha, R. Teixeira, and C. Diot, "Measuring and characterizing end-to-end route dynamics in the presence of load balancing," in *Proc. PAM*, 2011, pp. 235–244.
[31] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with rocketfuel," in *Proc. ACM SIGCOMM*, 2002, pp. 133–145.
[32] Y. Liao, W. Du, P. Geurts, and G. Leduc, "Decentralized prediction of end-to-end network performance classes," in *Proc. ACM CoNEXT*, 2011, Art. no. 14.
[33] *OpenFlow Switch Specification 1.3*. [Online]. Available: https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.3.0.pdf
[34] *PicOS Open vSwitch Configuration Guide*. [Online]. Available: https://docs.pica8.com/display/PicOS211sp/PicOS+Open+vSwitch+Configuration+Guide
[35] *Centec V580 OpenFlow Configuration Guide*. [Online]. Available: http://www.centecnetworks.com/en/DownView.asp?ID=1528&SortID=153
[36] *NoviKit 250 Efficient Flow Management Switch*. [Online]. Available: http://www.nvc.co.jp/pdf/product/noviflow/NoviKit250Datasheet.pdf
[37] X. Wang, M. Malboubi, S. Wang, S. Xu, and C.-N. Chuah, "ProgLIMI: Programmable LInk metric identification in software-defined WANs," Tech. Rep. [Online]. Available: http://faculty.uestc.edu.cn/wangxiong/en/lwcg/149769/content/7970.htm
[38] M. H. Firooz and S. Roy, "Link delay estimation via expander graphs," *IEEE Trans. Commun.*, vol. 62, no. 1, pp. 170–181, Jan. 2014.
[39] R. Z. Farahani and M. Hekmatfar, *Facility Location: Concepts, Models, Algorithms and Case Studies*. Berlin, Germany: Springer-Verlag, 2009.

**Xiong Wang** is currently an Associate Professor with the School of Information and Communication, University of Electronic Science and Technology of China, Chengdu, China. His research interests include network measurement and management, modeling and optimization, and algorithm design.

**Mehdi Malboubi** received the M.Sc. degree in computer science and the Ph.D. degree in electrical and computer engineering from the University of California at Davis.

**Zhihao Pan** is currently a Software Engineer with the NETEASE, Inc.

**Jing Ren** is currently a Lecturer with the School of Information and Communication, University of Electronic Science and Technology of China, Chengdu, China.

**Sheng Wang** is currently a Professor with the School of Information and Communication, University of Electronic Science and Technology of China, Chengdu, China. His research interests include network optimization and network architecture.

**Shizhong Xu** is currently a Professor with the School of Information and Communication, University of Electronic Science and Technology of China, Chengdu, China. His research interests include Internet of things and network science.

**Chen-Nee Chuah** (F'15) is currently a Professor in electrical and computer engineering with the University of California at Davis, Davis. Her research interests include Internet measurements, network management, and applying data and network science techniques to online social networks, security detection, digital healthcare, and intelligent transportation systems. She is a fellow of the IEEE and an ACM Distinguished Scientist. She was a recipient of the NSF CAREER Award and was named a Chancellor's Fellow of UC Davis in 2008.