Preferential Link Tomography: Monitor Assignment for Inferring Interesting Link Metrics

Yi Gao†, Wenbin Wu†, Wei Dong*†, Chun Chen†, Xiang-Yang Li‡, Jiajun Bu†
†Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science, Zhejiang University, China
‡Department of Computer Science, Illinois Institute of Technology, USA
{gaoyi, wuwb, dongw, chenc, bjj}@zju.edu.cn, xli@cs.iit.edu

***Abstract*—We study the problem of identifying additive and static link metrics of a set of *interesting links* in a communication network, by using end-to-end cycle-free path measurements among selected monitors. To uniquely identify the metrics of these interesting links, three questions should be addressed: *monitor assignment* (which nodes should serve as monitors), *paths selection* (which cycle-free paths connecting each pair of monitors will be used), and *link metric calculation*. Since assigning a node as a monitor usually requires non-negligible operational cost, we focus on assigning the minimum number of monitors (i.e., optimal monitor assignment) to identify all interesting links. By modeling the network as a connected graph, we propose *Scalpel*, an efficient preferential link tomography approach. Scalpel trims the original graph by a two-stage graph trimming algorithm and reuses existing method to assign monitors in the trimmed graph. We theoretically prove Scalpel has several key properties: 1) the graph trimming algorithm in Scalpel is minimal in the sense that further trimming the graph cannot reduce the number of monitors; 2) the obtained assignment is able to identify all interesting links in the original graph; and 3) an optimal monitor assignment in the graph after trimming is also an optimal monitor assignment in the original graph. Extensive simulations based on both synthetic topologies and real network topologies show the effectiveness of Scalpel. Compared with state-of-the-art, our approach reduces the number of monitors by 39.0%∼98.6% when 50%∼1% of all links are interesting links.**

I. Introduction

Inferring fine-grained network characteristics using aggregated measurements, as known as *network tomography* [1], is an effective technique to facilitate various network operations [2], [3], [4], such as network monitoring, load balance, and fault diagnosis. In communication networks, a subset of nodes with monitoring capabilities, i.e., *monitors*, can initiate and collect end-to-end measurements of selected cycle-free paths. Then, by using these *end-to-end* metrics, network tomography techniques can decompose them to *hop-by-hop* link metrics by solving a system of equations.

In many cases, these link metrics are additive [2], [5]. For example, delay is a typical additive metric since an end-to-end path delay is the sum of all link delays, while a multiplicative metric like packet delivery ratio can be expressed in an additive form by applying the $\log(\cdot)$ function. In order to identify additive link metrics, we need to solve a linear system, where the unknown variables are the link metrics, and the known constants are the end-to-end path measurements, each equal to the sum of the corresponding link metrics along a path [2]. Since these end-to-end path measurements are conducted among monitors, the monitor assignment (which nodes should be assigned as monitors) becomes a key problem. On one hand, the monitor assignment should comply with certain conditions to enable a sufficient number of linearly independent measurements. For example, assume v is a node with degree two in a network and l_1, l_2 are two links that incident to v. If node v is not assigned as a monitor, no matter how to conduct measurement paths among other monitors, we can only calculate the sum of the link metrics of l_1, l_2, instead of their individual link metrics. On the other hand, we usually want to minimize the number of monitors assigned, since assigning a node as a monitor usually needs non-negligible operational cost (e.g., hardware/software, human efforts). Ma et al. [2] successfully solved the above problem by proposing the MMP algorithm which identifies *all* link metrics in the network by assigning the *minimum* number of monitors.

Unfortunately, inferring all link metrics may incur a high overhead and is not necessary in many applications. For example, in a network from the Rocketfuel project [6], 117 out of 182 nodes should be assigned as monitors to identify all link metrics. In practice, network managers are usually interested in a subset of link metrics, instead of all links in the network. For example, in the Internet, some links (e.g., problematic links reported by customers or links located in critical infrastructures such as hospitals and fire departments) are known to be more important than other links. In sensor networks [7], [8], [9], links near the basestation are more important since they usually carry a large volume of traffic. By identifying those interesting links (i.e., preferential links), we can significantly reduce the monitoring overhead.

Given a network topology and a set of *interesting links*, how to assign a minimum number of monitors to identify these interesting link metrics is challenging due to the following two reasons. First, due to the complex network topology, a monitor can help identify a link far away from it. This precludes algorithms like divide-and-conquer. Second, it is difficult to obtain the dependency between the identifiability of a link and a particular monitor. Thus, it is difficult to implement a naive method that removes the monitors on which no interesting links depend.

*Corresponding author. We gratefully acknowledge our shepherd John Chi Shing Lui and the ICNP reviewers. This work is supported by the National Science Foundation of China Grant No. 61202402, the Fundamental Research Funds for the Central Universities, National Key Technology R&D Program (2012BAI34B01), the Research Fund for the Doctoral Program of Higher Education of China (20120101120179), Demonstration of Digital Medical Service and Technology in Destined Region.

As a first step towards addressing the monitor assignment problem of preferential network tomography, we propose Scalpel which carefully trims the original network graph without sacrificing the optimal solution and use existing method to assign monitors. Although it is difficult to obtain the dependency between link identifiability and a particular monitor, we are able to effectively narrow down the search space by safely trimming unrelated network components. After graph trimming, Scalpel reuses the MMP algorithm [2] for monitor assignment. The preferential link tomography problem studied in this paper generalizes the problem studied in MMP in which all links are considered as interesting links. It is more flexible and practical to be able to assign monitors with any given interesting links. Scalpel has several salient features. The graph trimming algorithm in Scalpel is minimal in the sense that further trimming the graph cannot reduce the number of monitors that should be assigned to identify all interesting links (formally given by Theorem VI.3). As the correctness of Scalpel, we can prove that Scalpel can identify all interesting links in the original graph, including those trimmed links (formally given by Theorem VI.1). Scalpel also opens the door for the minimum number of monitor assignment due to the following property: an optimal monitor assignment in the graph after trimming is also an optimal monitor assignment in the original graph (formally given by Theorem VI.2).

We implement Scalpel and evaluate it through extensive simulations based on both synthetic topologies and real network topologies. The time complexity of Scalpel is linear in terms of the number of vertices and links, making it be able to assign monitors in large scale networks efficiently. Results show that when 50%$\sim$1% of all links are interesting links, Scalpel reduces the number of monitors by 42.7%$\sim$98.6% and 39%$\sim$96% in synthetic topologies and real network topologies.

The contributions of this paper are summarized as follows.

- We are the first to identify the preferential link tomography problem to efficiently infer the metrics of a set of interesting links.
- We propose Scalpel, an efficient preferential link tomography approach. Scalpel trims the original graph by a two-stage graph trimming algorithm and reuses existing method to assign monitors in the trimmed graph. We prove that the graph trimming algorithm in Scalpel is minimal in the sense that further trimming the graph cannot reduce the number of monitors that should be assigned to identify all interesting links. We also prove that the obtained assignment is able to identify all interesting links in the original graph.
- We prove that an optimal monitor assignment in the graph after trimming is also an optimal monitor assignment in the original graph. This property opens the door for the minimum number of monitor assignment based on our graph trimming algorithm.
- We implement and evaluate Scalpel by extensive simulations. The time complexity of Scalpel is linear in terms of the number of vertices and links, making it be able to trim large scale networks efficiently. Results shows that Scalpel is able to reduce the number of monitors by 39.0%$\sim$98.6% when 50%$\sim$1% of all links are interesting links.

The rest of the paper is organized as follows. Section II discusses the related work. Section III presents the problem formulation. Section IV gives some graph theory fundamentals used in this paper. Section V describes Scalpel in detail. Section VI theoretically analyzes Scalpel and proves its three important properties. Section VII presents the evaluation of Scalpel, and finally, Section VIII concludes this paper.

II. Related Work

Knowledge of the internal state of a network (e.g., link delays) is essential for network monitoring, fault diagnosis, load balancing and other network operations. In order to measure the network metrics, different approaches have been proposed in the literature. The first category includes *hop-by-hop* approaches, which use diagnostic tools such as *traceroute*, *pathchar* [10], and *Network Characterization Service (NCS)* [11] to measure hop-by-hop link metrics directly. By sending multiple probes with different time-to-live (TTL) fields, *traceroute* can measure the delay of each hop on the probed path. *Pathchar* uses a similar approach to measuring hop-by-hop delays, capacities and loss rates. *NCS* also reports available capacity of each hop. These approaches require that the Internet Control Message Protocol (ICMP) be supported at all nodes. Further, the above tools send a relatively large number of probes, introducing non-negligible overhead.

The other category includes end-to-end approaches, which use end-to-end metrics to infer internal link metrics [3], [12], [4], [13], [2], [14], as known as network tomography. These approaches utilize the path information to calculate link metrics, reducing the number of probes significantly. They assume the network is controllable, i.e., a monitor is able to send measurement packets with pre-determined paths. This controllable network assumption is reasonable since it is generally supported for networks under common administration [2]. A key problem is how to assign the minimum number of monitors so that the operational cost can be reduced. The basic idea is to build a linear system from the path measurements and use linear algebraic techniques to calculate the unknown link metrics [15], [16]. As shown in [15], the problem is challenging due to the existence of linearly dependent paths. Several approaches [16], [17], [2] have been used to calculate the link metrics. When the link metrics are binary variables (e.g., normal or failed), Chen et al. give a topology requirement to identify all failed links using only one monitor measuring cycles. Many approaches focus on the usual case when the link metrics are arbitrary valued. When cyclic measurement paths are allowed, Gopalan et al. [5] give the necessary and sufficient conditions on the network topology. Since routing along cycles is typically prohibited in real networks, cycle-free measurement path are preferred. Ma et al. [2] give the necessary and sufficient conditions on the network topology when only cycle-free measurement paths are used. In order to identify all links in a connected network, Ma et al. also propose an efficient algorithm called MMP [2] to assign the minimum number of monitors as well as an efficient path construction algorithm [18]. Different with [2], we consider a more practical and general case when we

are only interested in a subset of links. Assigning monitors to identify interesting links faces non-trivial challenges due to the complex dependency between the link identifiability and each monitor.

III. Problem Formulation

This section gives the formulation of the minimum monitor assignment problem for identifying a set of interesting links.

A. Network Model and Assumptions

We assume that the network topology is known and does not change during the measurement period. We model the network topology as an undirected graph $\mathcal{G} = (V(\mathcal{G}), L(\mathcal{G}))$, where $V(\mathcal{G})$ and $L(\mathcal{G})$ are the sets of vertices and links, respectively. Without loss of generality, we can assume that graph $\mathcal{G}$ is connected, since different connected components of the network can be monitored separately. We denote the link that incidents to vertices u and v by uv. We assume the links are symmetric, i.e, the link metrics of uv and vu are the same. We further assume that there is no self-loop link in $L(\mathcal{G})$, and there is at most one link connecting two vertices. A set $\mathcal{I} \subseteq L(\mathcal{G})$ is a set of interesting links. A subset of vertices in $V(\mathcal{G})$ are assigned as monitors and can initiate/collect end-to-end measurements for identifying the metrics of links in $\mathcal{I}$. Since routing along cycles is typically prohibited in real networks [2], cycle-free measurement paths among monitors are preferred.

B. Problem Formulation

A monitor m_A can initiate a measurement packet to another monitor m_B through a *simple path*. A simple path does not contain repeated vertices. We assume that monitors can control the routing of the measurement packet. Such routing is generally supported in common networks like single-ISP networks and captures capabilities of new generation of networks performing Software-Defined Networking (SDN) [2]. Monitor m_B can obtain the path measurement, which is the sum of all link metrics along the path $\mathcal{P}$. In order to simplify the presentation, we use l (or $\mathcal{P}$) to denote both the link (or path) and its link (or path) metric. γ path measurements are obtained by sending measurement packets among monitors. Then we can build a linear system to identify the interesting links. Let $n = |L|$ be the number of all links in L, $\mathbf{l} = (l_1, ..., l_n)^T$ be the column vector of all link metrics, and $\mathbf{p} = (\mathcal{P}_1, ..., \mathcal{P}_\gamma)^T$. The relationship between $\mathbf{l}$ and $\mathbf{p}$ can be formulated as the following linear system.

$$\mathbf{Rl} = \mathbf{p}, \tag{1}$$

where $\mathbf{R} = (R_{ij})$ is a $\gamma \times n$ *measurement matrix*. $R_{ij} \in \{0, 1\}$ denotes whether link l_j is in measurement path $\mathcal{P}_i$. The minimum monitor assignment problem is how to find the minimum number of monitors such that there exists a measurement matrix $\mathbf{R}$ which makes all interesting link metrics in $\mathbf{l}$ be solvable in the above linear system. If the link metric can be calculated from the linear system obtained by a monitor assignment, this link is *identifiable* by the monitor assignment.

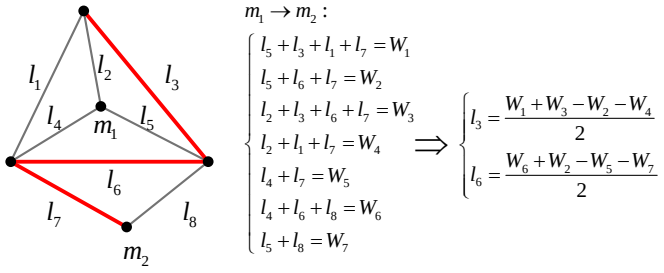


Fig. 1. A toy example. Two monitors are used to identify link metrics. In this example, the metrics of two interesting links l_3, l_6 can be identified by solving the linear system in the right part. However, the other interesting link l_7 cannot be identified if only these two monitors are assigned.

C. A Toy Example

Figure 1 shows a toy network with five vertices and eight links (l_1 to l_8). Among these links, three of them are interesting links (i.e., $\mathcal{I} = \{l_3, l_6, l_7\}$). In order to identify the interesting link metrics, we conduct seven path measurements from m_1 to m_2. By solving the linear system shown in the figure, we can identify the link metrics of l_3 and l_6. However, the link metric of l_7 cannot be identified in this example, no matter how we conduct path measurements between the two monitors. In fact, two monitors are not sufficient to identify all the three interesting links, no matter how we assign monitors in the network. Given the monitor assignment $\{m_1, m_2\}$, links l_3, l_6 are identifiable while link l_7 is not identifiable. In the following sections, we will describe our method Scalpel to perform efficient preferential link tomography.

IV. Fundamentals on Monitor Assignment

We first introduce several concepts in graph theory, which are used in this paper.

- A graph is *connected* when there exists at least one path from any vertex to any other vertex in the graph.
- A graph (or component) is *bi-connected* when the graph is still connected after removing one arbitrary vertex and the links incident to it. It includes at least two vertices. We also call a graph with one link and its two endpoints as a bi-connected graph (or component). Note that partitioning a connected graph into bi-connected components is a classical graph theory problem which can be solved in linear time [19].
- A graph (or component) is *tri-connected* when the graph is still connected after removing any two vertices and the links incident to them. It includes at least three vertices. A triangle is also treated as a tri-connected component in this paper. Note that there exists a classical graph theory algorithm called SPQR-tree [20] which can partition a bi-connected graph into a number of tri-connected components and/or circles in linear time.
- A *1-cut-v* for a connected graph $\mathcal{G}$ is a vertex whose removal will disconnect the graph $\mathcal{G}$.
- A pair of *2-cut-vs* for a connected graph $\mathcal{G}$ are two vertices that removing one of them does not

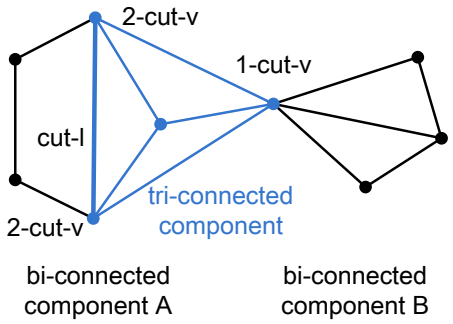


Fig. 2. A sample graph to illustrate some graph theory concepts used in this paper. The 1-cut-v separates the graph into two bi-connected components. The two 2-cut-vs separate one bi-connected components into one tri-connected component and a circle. The link cut-l connects the two 2-cut-vs.

disconnect $\mathcal{G}$, but removing both disconnects $\mathcal{G}$.

- A *sep-v* is a separating vertex which could be a 1-cut-v, or a 2-cut-v, or both.
- A *cut-l* is a link which connects a pair of 2-cut-vs.

Figure 2 shows an example with two bi-connected components separated by one 1-cut-v. The left bi-connected component A is further partitioned into one tri-connected component and one circle by a pair of 2-cut-vs. The link that connects to the two 2-cut-vs is a cut-l.

Then we give some important results from existing work [2], [21].

Theorem IV.1. *If $\mathcal{G}$ is a tri-connected graph, all of its link metrics are identifiable by assigning any three monitors.*

Corollary IV.2. *If $\mathcal{T}$ is a tri-connected component, all of its link metrics are identifiable by assigning any three vertices in the original graph $\mathcal{G}$ as monitors, when the three vertices v_1, v_2, v_3 satisfy one of the following conditions. 1) v_1, v_2, v_3 are in $\mathcal{T}$; 2) two of them are in $\mathcal{T}$ (not sep-v) and one is not in $\mathcal{T}$; 3) one of them is in $\mathcal{T}$ (not sep-v), two vertices v_i, v_j are not in $\mathcal{T}$ but in the same bi-connected component as $\mathcal{T}$.*

Definition IV.1. *For a tri-connected graph $\mathcal{G}$ (or component) and two vertices v_1, v_2 in it, its **interior links** are defined as links that do not incident to either of the two vertices; and its **exterior links** are defined as links that incident only one vertex (v_1 or v_2).*

Theorem IV.3. *For a tri-connected component $\mathcal{T}$ with two monitors assigned in it (or two sep-vs which connect to two monitors through two paths without repeated vertices outside $\mathcal{T}$, or one monitor in $\mathcal{T}$ and one such sep-v), all its interior links are identifiable and all its exterior link is unidentifiable.*

Figure 3 shows a typical tri-connected component $\mathcal{T}$ and two vertices m_1, m_2 which connects to v_1, v_2 through two paths. Theorem IV.1 says if we assign three monitors in $\mathcal{T}$ (e.g., v_1, v_2, m_3), all links in $\mathcal{T}$ are identifiable. Corollary IV.2 says if we assign m_1, m_2, m_3 as monitors, all links in $\mathcal{T}$ are still identifiable. Link l_2 is an interior link w.r.t. vertices v_2, m_3. Link l_1, l_3, l_4 and l_5 are exterior links w.r.t. vertices v_2, m_3. Theorem IV.3 says if we assign two monitors v_2, m_3, the interior links (i.e. l_2) can be identified and the exterior links

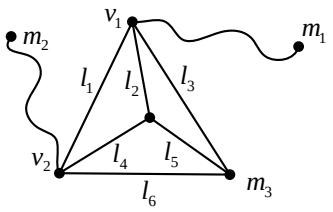


Fig. 3. A typical tri-connected graph to illustrate several results from previous works. l_2 is an interior link w.r.t. vertices v_2, m_3. And l_2 can be identified by two monitors $\{v_2, m_3\}$ or $\{m_2, m_3\}$.

(i.e. l_1, l_3, l_4, l_5) cannot be identified. Note that l_6 is not an exterior link since it incidents both of the two vertices v_2, m_3. Another example is that if we assign m_2, m_3 as two monitors, the interior links (i.e. l_2) can still be identified and the exterior links (i.e. l_1, l_3, l_4, l_5) cannot be identified.

V. SCALPEL

In this section, we describe the proposed Scalpel algorithm in detail. First, Scalpel trims the original graph into a trimmed graph. Then Scalpel assigns monitors in the trimmed graph. Figure 4 shows the overview of Scalpel. Scalpel trims the original graph $\mathcal{G}$ in two stages. The first stage trims some bi-connected components in $\mathcal{G}$ and outputs a intermediate trimmed graph $\mathcal{G}_{t_1}$. The second stage trims some SPQR components (i.e., tri-connected components or circles) in the graph $\mathcal{G}_{t_1}$ and outputs a trimmed graph $\mathcal{G}_t$ and a set of *helper* vertices $\mathcal{H}$. $\mathcal{H}$ is a subset of $V(\mathcal{G} - \mathcal{G}_t)$. Detailed information about choosing these helper vertices will be given in Section V.C. Based on the trimmed graph $\mathcal{G}_t$, Scalpel assigns monitors to identify interesting links in the original graph $\mathcal{G}$.

A. Definitions

Before describing Scalpel in detail, we first give the definitions of a *monitor assignment* and an *optimal monitor assignment* formally.

Definition V.1. *A monitor assignment $\mathcal{M}(\mathcal{G}_x)$ is a subset of vertex set $V(\mathcal{G}_x)$, in which each vertex is assigned as a monitor. Here, $\mathcal{G}_x$ represents any graph, such as the original graph $\mathcal{G}$ and the trimmed graph $\mathcal{G}_t$.*

In particular, $\mathcal{M}(\mathcal{G}_t \cup \mathcal{H})$ is a set of vertices in $V(\mathcal{G}_t) \cup \mathcal{H}$ which are assigned as monitors. Figure 5 gives an example. The dotted part of the graph is trimmed and the solid part is the trimmed graph $\mathcal{G}_t$. A helper vertex v_7 is in the dotted part. A monitor assignment $\mathcal{M}(\mathcal{G}_t \cup \mathcal{H})$ is $\{v_4, v_7\}$. Since we want to assign minimum number of monitors, we have the following definition about optimal monitor assignment.

Definition V.2. *An optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_x)$ is a monitor assignment $\mathcal{M}(\mathcal{G}_x)$, in which the number of monitors is **minimum** for identifying all the interesting links in the **original graph** $\mathcal{G}$.*

It is worth noting that an optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_x)$ is able to identify all interesting links in the **original graph** $\mathcal{G}$, instead of the graph $\mathcal{G}_x$. According to this definition, we have the following two optimal monitor assignments: 1)

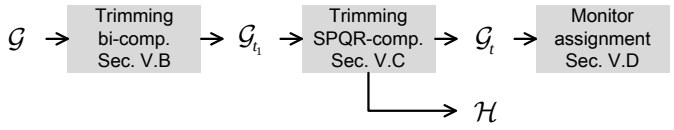


Fig. 4. Overview of Scalpel, which first trims the graph in two stages and assign monitors in the trimmed graph.

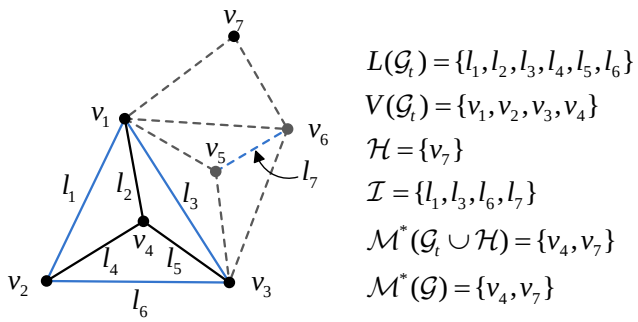


Fig. 5. An example that illustrates a monitor assignment $\mathcal{M}(\mathcal{G}_t \cup \mathcal{H}) = \{v_4, v_7\}$. In order to identify the four interesting links (l_1, l_3, l_6, l_7), assigning v_4 and v_7 as monitors are optimal.

optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ in $V(\mathcal{G}_t) \cup \mathcal{H}$ (i.e., vertices in the trimmed graph $\mathcal{G}_t$ and the helper vertices set $\mathcal{H}$) , and 2) the optimal monitor assignment $\mathcal{M}^*(\mathcal{G})$ in $V(\mathcal{G})$. The difference between $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ and $\mathcal{M}^*(\mathcal{G})$ is that the former can only assign vertices in $V(\mathcal{G}_t) \cup \mathcal{H}$ as monitors and the latter can assign vertices in $V(\mathcal{G})$ as monitors. Since $(V(\mathcal{G}_t) \cup \mathcal{H}) \subseteq V(\mathcal{G})$, $\mathcal{M}^*(\mathcal{G})$ has more choices as monitors. In Figure 5, $V(\mathcal{G}_t) \cup \mathcal{H} = \{v_1, v_2, v_3, v_4, v_7\}$ and $V(\mathcal{G}) = \{v_1, v_2, ..., v_7\}$. Assigning v_4 and v_7 as monitors is able to identify all the interesting links in $\mathcal{I}$ $(= \{l_1, l_3, l_6, l_7\})$ in the original graph. In this example, $\{v_4, v_7\}$ is both an optimal monitor assignment $\mathcal{M}^*(\mathcal{G})$ in $V(\mathcal{G})$ and an optimal assignment $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ in $V(\mathcal{G}_t) \cup \mathcal{H}$. Note that these two kinds of optimal monitor assignments are not unique. That is, different assignments which are able to identify all interesting links may have the same number of monitors, which are also optimal.

B. Trimming Bi-connected Components

The first stage graph trimming iteratively trims bi-connected components with no interesting links. Algorithm 1 gives the first stage graph trimming algorithm. The input of this algorithm is a connected graph $\mathcal{G} = \{V(\mathcal{G}), L(\mathcal{G})\}$ and a set $\mathcal{I} \subseteq L(\mathcal{G})$ of interesting links. Scalpel first partitions the graph $\mathcal{G}$ into bi-connected components (line 1). Then Scalpel inserts all bi-connected components with no interesting links into a queue (line 2∼5). Then Scalpel iteratively trims bi-connected components which only have one 1-cut-v and have no interesting links (line 6∼13). The first stage graph trimming only needs to traverse the graph once. Further, partitioning a connected graph into bi-connected components can be done in linear time [19]. Therefore, the time complexity of the first stage graph trimming is $O(|V(\mathcal{G})| + |L(\mathcal{G})|)$.

Figure 6 shows an example. Five bi-connected components ($\mathcal{B}_1$ to $\mathcal{B}_5$) are separated by four 1-cut-vs (v_1 to v_4). We assume that only bi-connected component $\mathcal{B}_4$ and $\mathcal{B}_5$ have interesting

Algorithm 1 First Stage Graph Trimming

Input: A connected graph $\mathcal{G}$, a set $\mathcal{I}$ of interesting links
Output: The trimmed graph $\mathcal{G}_{t_1}$

1: partition $\mathcal{G}$ into bi-connected components $\mathcal{B}_1$, $\mathcal{B}_2$, ...
2: Let *Queue* be a queue for bi-connected components
3: **for** each bi-connected component $\mathcal{B}_i$ **do**
4: **if** $\{l | l \in \mathcal{B}_i, l \in \mathcal{I}\}=\emptyset$ and $\mathcal{B}_i$ has one 1-cut-v **then**
5: *Quene*.enqueue($\mathcal{B}_i$)
6: **while** *Queue*.notEmpty() **do**
7: $\mathcal{B} \leftarrow$ *Queue*.dequeue()
8: Let $\mathcal{B}_n$ be a neighbor bi-connected component of $\mathcal{B}$
9: delete $\mathcal{B}$ except the 1-cut-v
10: **if** the 1-cut-v is only in $\mathcal{B}_n$ **then**
11: mark the 1-cut-v as not a cut vertex
12: **if** $\{l | l \in \mathcal{B}_n, l \in \mathcal{I}\}=\emptyset$ and $\mathcal{B}_n$ has one 1-cut-v **then**
13: *Queue*.enqueue($\mathcal{B}_n$)

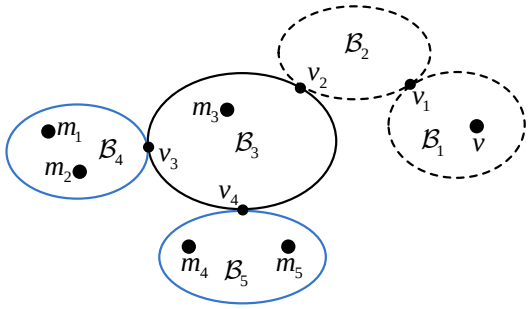


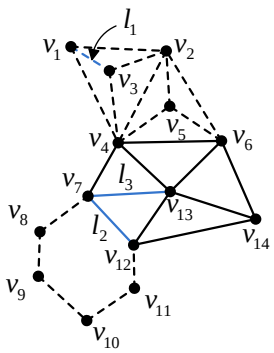
Fig. 6. An example to illustrate the first stage of graph trimming. Bi-connected components $\mathcal{B}_1$ and $\mathcal{B}_2$ without interesting links are trimmed in this stage.

links. In this case, Algorithm 1 will first insert $\mathcal{B}_1$ to the queue. Then $\mathcal{B}_1$ is trimmed and $\mathcal{B}_2$ is inserted to the queue. Then $\mathcal{B}_2$ is trimmed. Since $\mathcal{B}_3$ has two 1-cut-vs after v_2 is marked as not a cut vertex, $\mathcal{B}_3$ is not inserted into the queue. Finally, the solid part of the graph is left after the first stage graph trimming.

C. Trimming SPQR components

The second stage graph trimming trims some tri-connected components or circles (i.e., SPQR components) for each bi-connected component. Different with the first stage graph trimming, the SPQR components trimmed in this second stage may include some special interesting links. This links are *interior links* w.r.t. the two sep-vs s_1, s_2 in a tri-connected component $\mathcal{T}$. As defined by Definition IV.1 in Section IV, these interior links do not incident to either of the two sep-vs. We use int$(\mathcal{T}, s_1, s_2)$ to denote the set of interior links in $\mathcal{T}$ w.r.t. the two sep-vs s_1, s_2. Accordingly, we use ext$(\mathcal{T}, s_1, s_2)$ to denote the set of exterior links which incident one sep-v. Theorem IV.3 in Section IV says if we assign s_1, s_2 as monitors, all links in int$(\mathcal{T}, s_1, s_2)$ are identifiable and all links in ext$(\mathcal{T}, s_1, s_2)$ are not identifiable.

Algorithm 2 gives the second graph trimming algorithm. The input is the trimmed graph $\mathcal{G}_{t_1}$ after the first graph trimming stage and the interesting link set $\mathcal{I}$. The output is the trimmed graph $\mathcal{G}_t$, and a helper vertex set $\mathcal{H}$. Each help



$\mathcal{I} = \{l_1, l_2, l_3\}$
$V(\mathcal{T}_1) = \{v_1, v_2, v_3, v_4\}$
$V(\mathcal{T}_2) = \{v_2, v_4, v_5, v_6\}$
$V(\mathcal{T}_3) = \{v_4, v_6, v_7, v_{12}, v_{13}, v_{14}\}$
$V(\mathcal{T}_4) = \{v_7, v_8, v_9, v_{10}, v_{11}, v_{12}\}$
$\mathcal{H} = \{v_5, v_8\}$
$h(l_2) = v_8, h(v_4v_6) = v_5$
$h^{-1}(v_8) = l_2, h^{-1}(v_5) = v_4v_6$

Fig. 7. An example used to help describe Algorithm 2, the second stage graph trimming. The dotted part in the graph is trimmed in this stage.

vertex v is associated with a link l in $\mathcal{G}_t$. We use $h(l) = v$ (or $l = h^{-1}(v)$ since it is a one-to-one mapping) to denote this association. Since this algorithm is quite complicated, we use an example shown in Figure 7 to help describe the algorithm. The second stage trimming algorithm works as follows.

Line 1∼6: for each bi-connected component in $\mathcal{G}_{t_1}$, Scalpel first partitions it into a number SPQR components, then inserts all SPQR components with only two sep-vs s_1, s_2 into a queue. In Figure 7, a bi-connected component $\mathcal{G}_{t_1}$ is partitioned into four SPQR components: $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3$ and $\mathcal{T}_4$. Then these four components are inserted into the queue.

Line 7∼13: for each circle $\mathcal{T}$ in the queue, Scalpel checks whether there are interesting links in $L(\mathcal{T}) - s_1s_2$ (line 10). If there is no such interesting links, Scalpel chooses a vertex (not s_1, s_2) as a helper vertex for link s_1s_2. Then Scalpel deletes $\mathcal{T}$ except s_1, s_2. In Figure 7, the SPQR component $\mathcal{T}_4$ is deleted since it does not include any interesting link except the cut-l l_2. Then v_8 is chosen as a helper vertex for the cut-l l_2 ($h(l_2) = v_8$).

Line 14∼23: for each tri-connected component $\mathcal{T}$ without any interesting exterior links (line 14), there are three cases before deleting $\mathcal{T}$. 1) If there exists a helper vertex $h(l)$ of a link l in $L(\mathcal{T})$ and no interesting link incidents to the helper vertex, Scalpel associates the cut-l s_1s_2 with the helper vertex $h(l)$ (line 14∼16). 2) If there exists a vertex v in $V(\mathcal{T}) - \{s_1, s_2\}$ and no interesting link incidents to v, add v as a helper vertex and associates the cut-l s_1s_2 with v (line 17∼19). 3) If neither of the previous two cases holds, choose a vertex v in $V(\mathcal{T}) - \{s_1, s_2\}$ as a helper vertex and associates the cut-l s_1s_2 with v (line 20∼22). After adding the helper vertex associated with the cut-l s_1s_2, Scalpel deletes $\mathcal{T}$ except s_1, s_2. In Figure 7, two tri-connected components $\mathcal{T}_1, \mathcal{T}_2$ are deleted in two iterations since there are no interesting exterior links. In the first iteration, tri-connected component $\mathcal{T}_1$ meets the third case. Then v_3 is chosen as a helper vertex that associates the cut-l v_2v_4. In the second iteration, tri-connected component $\mathcal{T}_2$ meets the first case. Then v_5 is chosen as a helper vertex that associates the cut-l v_4v_6. Note that the helper vertex v_3 that associates with link v_2v_4 is deleted when v_2v_4 is deleted along with the tri-connected component $\mathcal{T}_2$.

Line 24∼29: if an SPQR component $\mathcal{T}$ is deleted, its neighboring SPQR components need some additional operations for the next iteration. Specifically, it is possible that some SPQR components have two 2-cut-vs (i.e., meets the condition in line 28) after the deletion of the SPQR component $\mathcal{T}$. Therefore, these SPQR components need to be inserted to the queue. In Figure 7, before the deletion of $\mathcal{T}_1$, $\mathcal{T}_2$ is not in the queue since it has three 2-cut-vs v_2, v_4, v_6. After $\mathcal{T}_1$ is deleted, $\mathcal{T}_2$ is inserted into the queue, since it has two 2-cut-vs v_4, v_6.

Algorithm 2 Second Stage Graph Trimming

Input: A graph $\mathcal{G}_{t_1}$, a set $\mathcal{I}$ of interesting links
Output: Trimmed graph $\mathcal{G}_{t_2}$ (i.e., $\mathcal{G}_t$), helper vertex set $\mathcal{H}$

1: **for** each bi-connected component $\mathcal{B}_i$ **do**
2: partition $\mathcal{B}_i$ into SPQR components $\mathcal{T}_1$,$\mathcal{T}_2$,...
3: Let *Queue* be a queue for SPQR components
4: **for** each SPQR components $\mathcal{T}_j$ **do**
5: **if** $\mathcal{T}_j$ has only two sep-v s_1, s_2 **then**
6: *Queue*.enqueue($\mathcal{T}_j$)
7: **while** *Queue*.notEmpty() **do**
8: $\mathcal{T} \leftarrow$ *Queue*.dequeue()
9: **if** $\mathcal{T}$ is a circle **then**
10: **if** $(L(\mathcal{T}) - s_1s_2) \cap \mathcal{I} = \emptyset$ **then**
11: $\mathcal{H} = \mathcal{H} \cup \{v\},\ v \in (V(\mathcal{T}) - \{s_1, s_2\})$
12: $h(s_1s_2) = v$
13: delete $\mathcal{T}$ except s_1s_2
14: **else if** ext$(\mathcal{T}, s_1, s_2) \cap \mathcal{I} = \emptyset$ **then**
15: **if** $\exists h(l) \in \mathcal{H}, l \in L(\mathcal{T}), h(l) \cap V(\mathcal{I}) = \emptyset$ **then**
16: $h(s_1s_2) = h(l)$
17: **else if** $\exists v \in (V(\mathcal{T}) - \{s_1, s_2\}), L(v) \cap \mathcal{I} = \emptyset$ **then**
18: $\mathcal{H} = \mathcal{H} \cup \{v\}$
19: $h(s_1s_2) = v$
20: **else**
21: $\mathcal{H} = \mathcal{H} \cup \{v\},\ v \in (V(\mathcal{T}) - \{s_1, s_2\})$
22: $h(s_1s_2) = v$
23: delete $\mathcal{T}$ except s_1s_2
24: **if** $\mathcal{T}$ is deleted **then**
25: **for** each $\mathcal{T}_n, |V(\mathcal{T}) \cap V(\mathcal{T}_n)| > 0$ **do**
26: **if** $s \in \{s_1, s_2\}$ are only in $\mathcal{T}_n$ **then**
27: mark s as not 2-cut-v
28: **if** $\mathcal{T}_n$ has two sep-v **then**
29: *Queue*.enqueue($\mathcal{T}_n$)

The second stage graph trimming needs to partition each bi-connected component into SPQR components and traverse a number of SPQR components (line 5, 28 in Algorithm 2). Since the SPQR partition can be done in linear time [20], the time complexity of the second stage graph trimming is $O(|V(\mathcal{G}_{t_1})| + |L(\mathcal{G}_{t_1})|)$, where $\mathcal{G}_{t_1}$ is the graph after the first stage graph trimming. Since the time complexity of the first stage graph trimming is also linear (Section V.B), the time complexity of the two-stage graph trimming is linear in terms of the number of vertices and links.

D. Monitor Assignment

After the two-stage graph trimming, Scalpel has a trimmed graph $\mathcal{G}_t$ and a set of helper vertices $\mathcal{H}$. Based on the trimmed graph $\mathcal{G}_t$, Scalpel assign monitors in $\mathcal{G}_t$ to identify interesting links in $\mathcal{G}_t$. We will prove that if a monitor assignment $\mathcal{M}(\mathcal{G}_t)$

can identify all links in $\mathcal{G}_t$, it can also identify interesting links in $\mathcal{G}$ (Theorem VI.1). This property of Scalpel enables us to assign monitors in the trimmed graph, which is usually much smaller than the original graph when there are a small number of interesting links. In the current implementation, Scalpel reuses an existing method called MMP to assign monitors in $\mathcal{G}_t$. MMP is able to assign the minimum number of monitors in a graph to identify *all* links. According to the above property, this assignment is able to identify the interesting links in the original graph $\mathcal{G}$.

Based on the output of graph trimming (the trimmed graph $\mathcal{G}_t$ and the helper vertices set $\mathcal{H}$), we will prove that an optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ in $V(\mathcal{G}_t) \cup \mathcal{H}$ has the same number of monitors as an optimal monitor assignment $\mathcal{M}^*(\mathcal{G})$ in $V(\mathcal{G})$ (Theorem VI.2). In other words, on optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ is also an optimal monitor assignment $\mathcal{M}^*(\mathcal{G})$. This property opens the door for designing optimal monitor assignment algorithm based on the trimmed graph $\mathcal{G}_t$ and the helper vertices set $\mathcal{H}$.

VI. Theoretical Analysis

In this section, we theoretically analyze Scalpel. In particular, we prove the following three theorems which describe the three important properties of Scalpel formally.

Theorem VI.1. *If a monitor assignment $\mathcal{M}(\mathcal{G}_t)$ in $V(\mathcal{G}_t)$ is able to identify all links in $\mathcal{G}_t$, it is also able to identify all interesting links in the original graph $\mathcal{G}$, including the interesting links been trimmed.*

Since Scalpel reuses existing method to assignment monitors in $\mathcal{G}_t$ to identify all links in it. According to this theorem, the obtained assignment by Scalpel is able to identify all interesting links in the original graph, including those trimmed links.

Theorem VI.2. *An optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ and an optimal monitor assignment $\mathcal{M}^*(\mathcal{G})$ have the same number of monitors.*

Theorem VI.2 says that an optimal monitor assignment in the graph after trimming is also an optimal monitor assignment in the original graph.

Theorem VI.3. *The graph trimming algorithm in Scalpel is minimal in the sense that if we further trim one SPQR component $\mathcal{T}$ from $\mathcal{G}_t$ and assign monitors in $\mathcal{G}_t - \mathcal{T}$ to identify all interesting links, the number of monitors cannot be reduced.*

A. Proof of Theorem VI.1

Proof. In the first stage graph trimming, every bi-connected component been trimmed does not include any interesting links. Therefore, we can focus on the interesting links been trimmed in the second stage.

In the second stage graph trimming, a number of SPQR components (circles or tri-connected components) are trimmed. According to line 9∼13 in Algorithm 2, a circle is only trimmed when there is no interesting link in it, except the link s_1s_2 connecting two sep-vs. Since the link s_1s_2 is not trimmed along with the circle (line 13), there are no interesting link in circles been trimmed. Then there is only one case left, which is the interesting links in the tri-connected components been trimmed. According to line 14∼23, it is possible that a tri-connected component with some interesting links are trimmed, when these interesting links are all *interior* links w.r.t. the two sep-vs s_1, s_2. We then prove that these interior interesting links in the tri-connected components been trimmed are identifiable when the condition given in the theorem holds.

Consider a bi-connected component $\mathcal{B}$ and one tri-connected component $\mathcal{T}$ been trimmed in $\mathcal{B}$. In the monitor assignment $\mathcal{M}(\mathcal{G}_t)$, there are at least two 1-cut-vs/monitors in $\mathcal{B}$ (otherwise no link in $\mathcal{B}$ is identifiable). Let two such vertices (i.e., 1-cut-vs or monitors) be v_1, v_2. Since $\mathcal{B}$ is a bi-connected component, we can find to paths without repeated vertices from these two vertices to s_1 and s_2, respectively. According to Theorem IV.3, all interior links in $\mathcal{T}$ are identifiable if we can find two paths without repeated vertices from s_1, s_2 to any two monitors. If v_i is a monitor, it meets the condition of Theorem IV.3. The other case is that v_i is a 1-cut-v, which separates the graph $\mathcal{G}_t$ into two parts, $\mathcal{G}_{ta}$ (which includes $\mathcal{T}$) and $\mathcal{G}_{tb}$. In this case, we prove the following claim by contradiction: we can always find a path in $\mathcal{G}_{tb}$ from the 1-cut-v v_i to a monitor.

Assume there is no monitor assigned in $\mathcal{G}_{tb}$, then no cycle-free measurement path includes links in $\mathcal{G}_{tb}$. Therefore, no links in $\mathcal{G}_{tb}$ can be identified, which contradicts the assumption of Theorem VI.1 that the assignment $\mathcal{M}(\mathcal{G}_t)$ is able to identify all links in $\mathcal{G}_t$. Therefore, the monitor assignment $\mathcal{M}(\mathcal{G}_t)$ includes at least one monitor assigned in $\mathcal{G}_{tb}$. Since $\mathcal{G}_{tb}$ is a connected graph component, we can always find a path from the 1-cut-v v_i to a monitor.

Therefore, all conditions of Theorem IV.3 are satisfied, which proves that all interesting links in $\mathcal{T}$ are identifiable by monitor assignment $\mathcal{M}(\mathcal{G}_t)$. □

B. Proof of Theorem VI.2

In order to prove Theorem VI.2, we need to first prove the following two theorems.

Theorem VI.4. *For any connected graph $\mathcal{G}$, the following inequality always holds.*

$$|\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})| \geq |\mathcal{M}^*(\mathcal{G})|. \tag{2}$$

where $|\mathcal{M}|$ is the number of monitors in the monitor assignment $\mathcal{M}$.

Theorem VI.5. *For any connected graph $\mathcal{G}$, the following inequality always holds.*

$$|\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})| \leq |\mathcal{M}^*(\mathcal{G})|. \tag{3}$$

Proof of Theorem VI.2. Combining inequalities 2 and 3 directly gives the following result.

$$|\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})| = |\mathcal{M}^*(\mathcal{G})|. \tag{4}$$

Therefore, Theorem VI.2 holds. □

Then, we prove Theorem VI.4 and Theorem VI.5.

1) Proof of Theorem VI.4:

PROOF. $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ and $\mathcal{M}^*(\mathcal{G})$ are both an optimal monitor assignment for identifying all interesting links in $\mathcal{G}$. The difference is that $\mathcal{M}^*(\mathcal{G})$ has more vertices as monitor candidates. Therefore, a solution of $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ is always a solution of $\mathcal{M}^*(\mathcal{G})$. Conversely, a solution of $\mathcal{M}^*(\mathcal{G})$ may not be a solution of $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$, since $\mathcal{M}^*(\mathcal{G})$ may include vertices in $V(\mathcal{G})$ but not in $V(\mathcal{G}_t) \cup \mathcal{H}$. Therefore, $\mathcal{M}^*(\mathcal{G})$ always includes less monitors than $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$, or includes the same number of monitors. Therefore, Theorem VI.4 holds. □

Theorem VI.4 is quite intuitive, since a larger feasible region (i.e., more monitor candidates) enables a better solution (i.e., less monitors at optimal).

2) Proof of Theorem VI.5: Compared with Theorem VI.4, Theorem VI.5 is counter-intuitive. In order to prove this theorem, we need to go through the two-stage graph trimming algorithm. For the first stage graph trimming, we first prove the following theorem.

Theorem VI.6. *Let V_s be a set of vertices which separate a connected graph $\mathcal{G}$ into two connected sub-graphs $\mathcal{G}_a$, $\mathcal{G}_b$. For each monitor assignment $\mathcal{M}(\mathcal{G})$ which is able to identify all interesting links in $\mathcal{G}_a$, the following claim always holds. If we keep the monitor assignment in $\mathcal{G}_a$ unchanged and change the monitor assignment in $\mathcal{G}_b$ into assigning all vertices in V_s as monitors, the new monitor assignment is still able to identify all interesting links in $\mathcal{G}_a$.*

PROOF OF THEOREM VI.6. As described in Section III, we can formulate the link metric identification by the following linear system.

$$\mathbf{Rl} = \mathbf{p}, \tag{5}$$

where $\mathbf{R}$ is the measurement matrix, $\mathbf{l}$ is a column vector of all links and $\mathbf{p}$ is a column vector of all metrics of measurement paths. Since graph $\mathcal{G}$ is separated into two connected sub-graphs $\mathcal{G}_a$, $\mathcal{G}_b$, we reorganize the link vector $\mathbf{l}$ into $(l_a^I, l_a^x, l_b^I, l_b^x)^T$, where the four parts are interesting links in $\mathcal{G}_a$, not interesting links in $\mathcal{G}_a$, interesting links in $\mathcal{G}_b$ and not interesting links in $\mathcal{G}_b$. The measurement matrix $\mathbf{R}$ and the vector $\mathbf{p}$ are also reorganized accordingly. Then the linear system becomes the following.

$$(R_a^I, R_a^x, R_b^I, R_b^x)(l_a^I, l_a^x, l_b^I, l_b^x)^T = (p_a^I, p_a^x, p_b^I, p_b^x)^T. \tag{6}$$

For a monitor assignment $\mathcal{M}(\mathcal{G})$ which is able to identify all interesting links in $\mathcal{G}_a$, its linear system has the following property. Let P be an invertible row transformation matrix that makes $P \cdot (R_a^I, R_a^x, R_b^I, R_b^x)$ be the reduced row echelon form. Then the coefficient matrix will be the following form.

$$P \cdot (R_a^I, R_a^x, R_b^I, R_b^x) = \begin{pmatrix} I & 0 & 0 & 0 \\ 0 & ... & ... & ... \\ 0 & ... & ... & ... \\ 0 & ... & ... & ... \end{pmatrix} \tag{7}$$

In the above reduced row echelon form, "I" is an identity matrix and "..." can be any sub-matrix.

Now, we change the monitor assignment in $\mathcal{G}_b$ into assigning all vertices in V_s as monitors. Then all measurement paths with vertices in V_s will be changed accordingly. The measurement matrix becomes $\mathbf{R}$'=$(R_a^I, R_a^x, 0, 0)$. Then we multiply the same row transformation matrix P by $\mathbf{R}$'.

$$P \cdot (R_a^I, R_a^x, 0, 0) = \begin{pmatrix} I & 0 & 0 & 0 \\ 0 & ... & 0 & 0 \\ 0 & ... & 0 & 0 \\ 0 & ... & 0 & 0 \end{pmatrix} \tag{8}$$

Therefore, the interesting links in $\mathcal{G}_a$ are still identifiable after changing the monitor assignment. □

Now we are ready to give the following theorem which describes how the first stage trimming affects the optimal monitor assignment.

Theorem VI.7. *An optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_{t_1})$ is also an optimal monitor assignment $\mathcal{M}^*(\mathcal{G})$.*

PROOF OF THEOREM VI.7. Without loss of generality, we assume $\mathcal{G}_{t_1}$ is the trimmed graph after trimming a single bi-connected component $\mathcal{B}$ from $\mathcal{G}$. If we can prove the theorem in this case, applying it recursively proves the theorem in a general case. According to Algorithm 1, the difference between $\mathcal{G}_{t_1}$ and $\mathcal{G}$ is that $\mathcal{G}$ has one bi-connected component $\mathcal{B}$ without any interesting links. $\mathcal{G}_{t_1}$ connects to $\mathcal{B}$ by a single 1-cut-v.

Then we prove the theorem by contradiction. Assume the monitor assignment $\mathcal{M}^*(\mathcal{G}_{t_1})$ is not an optimal monitor assignment $\mathcal{M}^*(\mathcal{G})$. Then there exists an assignment $\mathcal{M}(\mathcal{G}) \not\subseteq \mathcal{M}^*(\mathcal{G}_{t_1})$ which has less monitors than $\mathcal{M}^*(\mathcal{G}_{t_1})$ and can identify all interesting links. Therefore, this $\mathcal{M}(\mathcal{G})$ includes one or more monitors in $\mathcal{B}$ (not v since $v \in V(\mathcal{G}_{t_1})$), and includes less than $|\mathcal{M}^*(\mathcal{G}_{t_1})| - 1$ monitors in $\mathcal{G}_{t_1}$. Now we assign the 1-cut-v as a monitor instead of the monitors in $\mathcal{B}$. Let this new assignment be $\mathcal{M}'(\mathcal{G})$. According to Theorem VI.6, all interesting links in $\mathcal{G}_{t_1}$ are still identifiable. Since there are no interesting link in $\mathcal{B}$, this new monitor assignment $\mathcal{M}'(\mathcal{G})$ can identify all links in $\mathcal{G}$. Further, all monitors in $\mathcal{M}'(\mathcal{G})$ are in $\mathcal{G}_{t_1}$, so $\mathcal{M}'(\mathcal{G})$ is a monitor assignment $\mathcal{M}(\mathcal{G}_{t_1})$. Note that the new assignment $\mathcal{M}'(\mathcal{G}) = \mathcal{M}(\mathcal{G}_{t_1})$ has less than $|\mathcal{M}^*(\mathcal{G}_{t_1})| - 1 + 1 = |\mathcal{M}^*(\mathcal{G}_{t_1})|$ monitors. This contradicts to that $\mathcal{M}^*(\mathcal{G}_{t_1})$ is optimal in $V(\mathcal{G}_{t_1})$. □

Figure 8 shows an example of the above proof. A bi-connected component $\mathcal{B}$ is trimmed since it does not has any interesting links. v is the 1-cut-v separating $\mathcal{G}_{t_1}$ and $\mathcal{B}$. The optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_{t_1})$ includes four monitors v_1, v_2, v_3, v_4. Then we prove Theorem VI.7 by contradiction. Assume $\{v_1, v_2, v_3, v_4\}$ is not an optimal monitor assignment $\mathcal{M}^*(\mathcal{G})$. Then there exists an assignment $\mathcal{M}(\mathcal{G}) \not\subseteq \mathcal{M}^*(\mathcal{G}_{t_1})$ which has three or less monitors and is able to identify all interesting links. Let this assignment be $\{v_1, v_2, v_6\}$. Now consider a monitor assignment $\mathcal{M}'(\mathcal{G}) = \{v_1, v_2, v\}$. According to Theorem VI.6, this assignment is also able to identify all interesting links in $\mathcal{G}_{t_1}$. Since there is no interesting links in $\mathcal{B}$, assignment $\{v_1, v_2, v\}$ is able to identify all interesting links in $\mathcal{G}$. Further, m_1, m_2, v are all in $\mathcal{G}_{t_1}$, assignment $\{v_1, v_2, v\}$ is an optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_{t_1})$. This contradicts that $\{v_1, v_2, v_3, v_4\}$ is an optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_{t_1})$ since it has one more monitors than $\{v_1, v_2, v\}$.

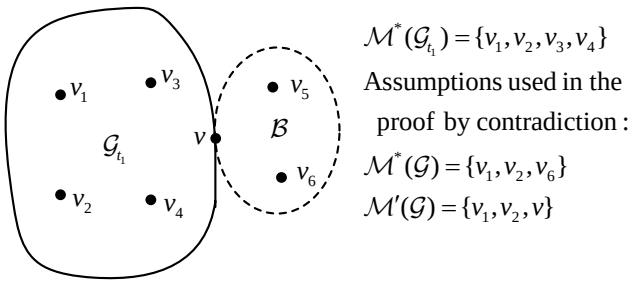


Fig. 8. An example that helps illustrate the proof of Theorem VI.7.

For the second stage, we first give the following theorem.

Theorem VI.8. *Let v_1, v_2 be a pair of 2-cut-vs which separate a connected graph $\mathcal{G}$ into two connected sub-graphs $\mathcal{G}_a$ and $\mathcal{G}_b$. For each monitor assignment $\mathcal{M}(\mathcal{G})$ which is able to identify all interesting links in $\mathcal{G}_a$ and $\mathcal{M}(\mathcal{G})$ includes a single monitor m in $V(\mathcal{G}_b) - \{v_1, v_2\}$, the following claim always holds. If we keep the monitor assignment in $\mathcal{G}_a$ unchanged and change the monitor assignment in $\mathcal{G}_b$ into assigning another vertex in $V(\mathcal{G}_b) - \{v_1, v_2, m\}$ as a monitor, the new monitor assignment is still able to identify all interesting links in $\mathcal{G}_a$.*

Due to the space limit, we omit the detailed proof of the above theorem. The proof of this theorem can be found in the technical report of this work [22].

Based on Theorem VI.8, we can prove the following theorem which describes how the second stage graph trimming affects the number of monitors in an optimal monitor assignment.

Theorem VI.9. *For any connected graph $\mathcal{G}$, the following inequality always holds.*

$$|\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})| \leq |\mathcal{M}^*(\mathcal{G}_{t_1})|. \tag{9}$$

PROOF OF THEOREM VI.9. Without loss of generality, we assume the second stage trimming only trims one single SPQR component $\mathcal{T}$. If the theorem holds in this case, applying it iteratively proves the theorem in a general case.

We start from an optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_{t_1})$ in the graph after the first stage trimming. If the monitor assignment $\mathcal{M}^*(\mathcal{G}_{t_1})$ does not include any monitors in $\mathcal{T}$, $\mathcal{M}^*(\mathcal{G}_{t_1}) = \mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$, and the theorem holds. Now we consider the case that $\mathcal{M}^*(\mathcal{G}_{t_1})$ includes monitors assigned in $\mathcal{T}$. Let the two 2-cut-vs be v_1, v_2 and the set of these monitors be $\mathcal{M}_\mathcal{T} = \mathcal{M}(\mathcal{T} - \{v_1, v_2\})$. Now we modify the optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_{t_1})$ into a new assignment $\mathcal{M}'$ according to the following rules. If there is one single monitor m in $\mathcal{M}_\mathcal{T}$, we assign the help vertex $h(v_1v_2)$ that associates with the cut-l v_1v_2 as a monitor instead of m. If there are two or more monitors in $\mathcal{M}_\mathcal{T}$, we assign v_1, v_2 as two monitors instead of any monitors in $\mathcal{M}_\mathcal{T}$. Note that this modification reduces monitors or keeps the same number of monitors ($|\mathcal{M}'| \leq |\mathcal{M}^*(\mathcal{G}_{t_1})|$). Then we will discuss these two cases.

According to Theorem VI.8, assigning $h(v_1v_2)$ as a monitor instead of any vertex in $\mathcal{T}$ is still able to identify all interesting links in $\mathcal{G}_t$. When $\mathcal{T}$ is circle, there is no interesting links in $L(\mathcal{T}) - v_1v_2$ (line 10 in Algorithm 2), which means the new assignment is able to identify all interesting links in $\mathcal{G}$. When $\mathcal{T}$ is a tri-connected component, it is possible that $\mathcal{T}$ includes a number of interesting interior links in int($\mathcal{T}, v_1, v_2$). Line 15$\sim$19 of Algorithm 2 assure that Scalpel will give priority to choose a vertex without any interesting link that incidents to it as a helper vertex. This means if the old monitor choice in $\mathcal{T}$, together with other monitors in $\mathcal{G}_t$, can identify all interesting links in $\mathcal{T}$, replacing it by the helper vertex is still able to identify all interesting links in $\mathcal{T}$. Therefore, assigning $h(v_1v_2)$ as a monitor instead of any vertex in $\mathcal{T}$ is still able to identify all interesting links in the original $\mathcal{G}$.

According to Theorem VI.6, assigning v_1, v_2 as two monitors instead of any vertices in $\mathcal{T}$ is still able to identify all interesting links in $\mathcal{G}_t$. When $\mathcal{T}$ is circle, there is no interesting links in $L(\mathcal{T}) - v_1v_2$ (line 10 in Algorithm 2), which means the new assignment is able to identify all interesting links in $\mathcal{G}$. When $\mathcal{T}$ is a tri-connected component with interesting interior links, all these interesting interior links can be identified by assigning v_1, v_2 as monitors (Theorem IV.3). Therefore, assigning v_1, v_2 as two monitors instead of any vertices in $\mathcal{T}$ is still able to identify all interesting links in the original graph $\mathcal{G}$.

Therefore, no matter how may monitors in $\mathcal{T}$, the new assignment $\mathcal{M}'$ is still able to identify all interesting links in $\mathcal{G}$. Since all monitors in $\mathcal{M}'$ are in $V(\mathcal{G}_t) \cup \mathcal{H}$, it is an monitor assignment $\mathcal{M}(\mathcal{G}_t \cup \mathcal{H})$ which can identify all interesting links in $\mathcal{G}$. Therefore, $|\mathcal{M}'| \geq |\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})|$. Since $|\mathcal{M}'| \leq |\mathcal{M}^*(\mathcal{G}_{t_1})|$, $|\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})| \leq |\mathcal{M}^*(\mathcal{G}_{t_1})|$. This proves the theorem. □

Proof of Theorem VI.5. Combining the results of Theorem VI.7 and Theorem VI.9, we can prove Theorem VI.5.

PROOF. Theorem VI.7 says that an optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_{t_1})$ is one optimal monitor assignment $\mathcal{M}^*(\mathcal{G})$. Therefore, the following equation holds.

$$|\mathcal{M}^*(\mathcal{G}_{t_1})| = |\mathcal{M}^*(\mathcal{G})|. \tag{10}$$

Combing the above equation with the result of Theorem VI.9, the following inequality holds.

$$|\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})| \leq |\mathcal{M}^*(\mathcal{G})|. \tag{11}$$

This proves the theorem. □

C. Proof of Theorem VI.3

PROOF. When$\mathcal{T}$ has no sep-v, which means $\mathcal{G}_t = \mathcal{T}$. In this case, trimming $\mathcal{T}$ will cause that no monitor will be assigned, and no interesting link can be identified. When $\mathcal{T}$ has one sep-v, which means it is the only component left in the bi-connected component $\mathcal{B}$ that includes $\mathcal{T}$. Then all interesting links in $\mathcal{B}$ will not be identifiable since it only has one sep-v and no monitor is assigned in it (similar to the proof of Theorem VI.1). There are two more cases of $\mathcal{T}$: $\mathcal{T}$ has two sep-vs or more than two sep-vs.

1) When $\mathcal{T}$ has two sep-vs (s_1, s_2), there are further two cases: $\mathcal{T}$ is a circle or tri-connected component. 1.1) When

TABLE I. MAIN RESULTS OF SCALPEL IN DIFFERENT NETWORKS WITH DIFFERENT NUMBER OF INTERESTING LINKS.

Network	|V(G)|	|L(G)|	1%		5%		10%		50%	
			|trim-B|	|trim-T|	|trim-B|	|trim-T|	|trim-B|	|trim-T|	|trim-B|	|trim-T|
BA	300	596	0/1	139/142	0/1	125.8/142	0/1	110.4/142	0/1	34/142
ER	297	668	18.8/20	28/29	18.8/20	24.8/29	16.8/20	22.8/29	10/20	6/29
RG	296	840	13.2/17	25/31.4	11/17	20.6/34	9.8/17	13.6/34.6	4/17	2.4/35.6
dK-1009	1009	3773	167/170	97.8/101	160.6/170	89.3/101	152.6/170	81.6/101	80.1/170	23.5/101
dK-2485	2485	9402	599/607	190.4/195	574.8/607	172.5/195	547.4/607	154.5/195	299.9/607	48/195
dK-3018	3018	11275	492.3/499	289.3/298	474.3/499	261.6/298	449.3/499	243.3/298	253.3/499	69.6/298
dK-5072	5072	19367	1267.6/1282	381.3/391	1213/1282	349.3/391	1151/1282	316/391	627/1282	88/391
AS1221	318	758	138.5/142	42.3/46.6	130.2/142	40.9/52.9	122/142	35.4/53	58.8/142	9.7/53
AS1239	604	2268	101.7/104	58.7/61.2	96.9/104	53.5/62.3	92.3/104	47.6/61.75	49.9/104	13.7/63.8
AS1755	172	381	26.7/28	34/35.2	24.8/28	30.7/35.6	23.4/28	26.1/35.55	12.4/28	7.8/36.8
AS3257	240	404	104.1/107	42.1/46	98.8/107	31.2/46.6	92.5/107	27.2/47.25	45.4/107	6.8/48.85
AS3356	624	5299	28.8/30	55.9/58	28/30	50.9/58	26.1/30	45.7/58	14.5/30	14.4/58
AS3967	201	434	36.6/38	45.2/47.7	34.7/38	41.2/48.5	32.7/38	34.6/48.45	16.9/38	11.1/51
AS6461	182	296	84.3/87	36.1/37.4	81.3/87	30.8/38.2	74.3/87	28.4/39.25	40.1/87	8/41
AS7018	631	2078	56.3/58	152.7/158	54.2/58	139.3/158	50.8/58	121.8/158	27.1/58	36.4/157.95

$\mathcal{T}$ is a circle, there must be at least one link l (not s_1s_2) in $\mathcal{T}$ is an interesting link (line 10 and 13 in Algorithm 2). If $\mathcal{T}$ is trimmed, the link l cannot be identified no matter how we assign monitors in $\mathcal{G}_t - \mathcal{T}$. The reason is that the link l has at least one endpoint v whose degree is two and is not a monitor. 1.2) When $\mathcal{T}$ is a tri-connected component, there must be an exterior link l in $\mathcal{T}$ is an interesting link (line 14 and 23 in Algorithm 2). According to Theorem VI.6 monitors outside $\mathcal{T}$ are equivalent to assigning s_1, s_2 as two monitors for identifying link l. Further, according to Theorem IV.3, the exterior link l cannot be identified by two monitors s_1, s_2 in $\mathcal{T}$. Therefore, if $\mathcal{T}$ is trimmed, the link l cannot be identified no matter how we assign monitors in $\mathcal{G}_t - \mathcal{T}$.

2) When $\mathcal{T}$ has more than two sep-vs, Scalpel will not assign any monitors in $\mathcal{T}$ even if it is not trimmed. The reason is that Scalpel uses MMP to assign monitors in $\mathcal{G}_t$ and MMP will ignore SPQR components with more than two sep-vs [2]. Therefore, trimming $\mathcal{T}$ cannot reduce the number of monitors directly. Then we discuss the monitor assignment in $\mathcal{G}_t - \mathcal{T}$ after trimming $\mathcal{T}$. There are two cases. 2.1) $\mathcal{G}_t - \mathcal{T}$ is still a connected graph. In this case, Scalpel will assign exactly the same monitors no matter whether $\mathcal{T}$ is trimmed. 2.2) $\mathcal{G}_t - \mathcal{T}$ is not a connected graph. In this case, Scalpel needs to use MMP separately in each connected component. Compared with graph $\mathcal{G}_t$, there are less possible measurement paths in $\mathcal{G}_t - \mathcal{T}$. Therefore, at least the same number of monitors should be assigned.

Based on the above analysis, if we further trim one SPQR component $\mathcal{T}$ from $\mathcal{G}_t$, the number of monitors cannot be reduce if we need to identify all interesting links. □

VII. EVALUATION

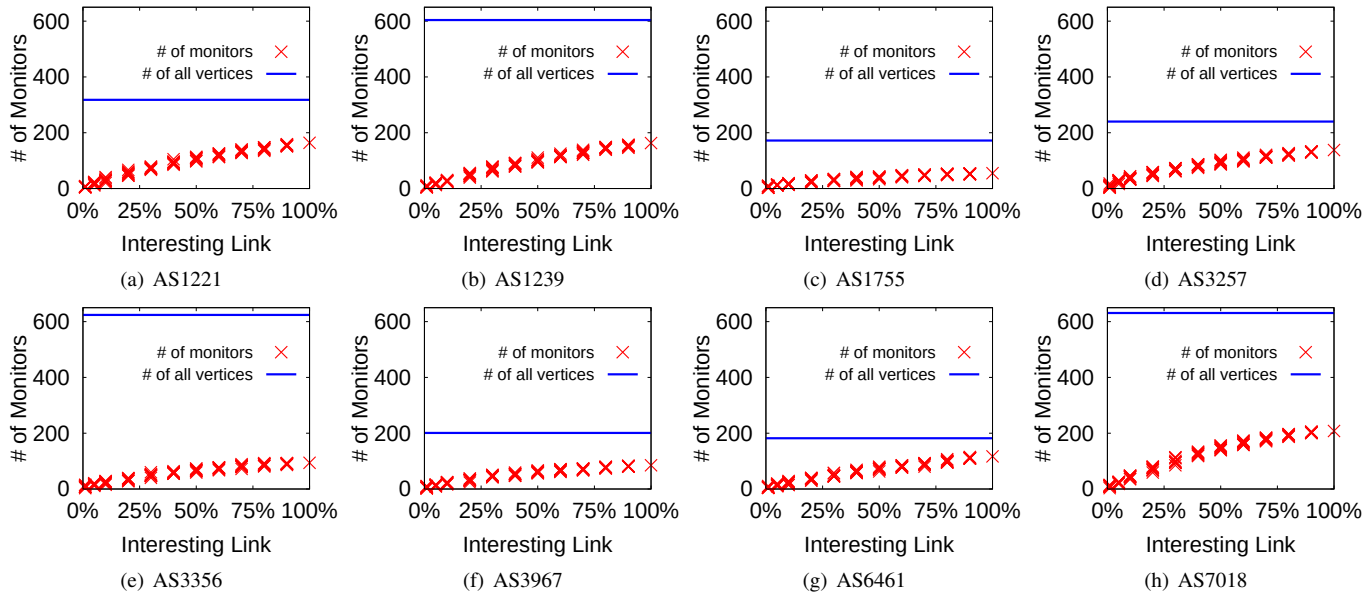
We evaluate Scalpel through extensive simulations on both synthetic network topologies and real network topologies. In this section, we will first give the evaluation methodology, including the evaluation metrics and detailed description of the topologies used. Then, we will show the graph trimming results and monitor assignment results of Scalpel.

A. Methodology

The first metric is the effectiveness of graph trimming. The first stage graph trimming of Scalpel trims a number of bi-connected components and the second stage graph trimming of Scalpel trims a number of SPQR components. Therefore, we report the number of bi-connected components and SPQR components which are trimmed when there are different number of interesting links. Then, we compare the monitor assignment results of Scalpel with MMP. The metrics used are the number of monitor assigned and the execution time.

For each network topology, we randomly choose different number of links as interesting links: 1%, 5%, 10%, 50% and 100% of all links in the original network. We ran each simulation 10 times and report the average values. We use both synthetic topologies and real network topologies.

1) Synthetic topologies. We consider four widely used synthetic topologies. *Barabási-Albert (BA) graph*. A Barabási-Albert (BA) graph is generated by the following steps. We begin with a small connected graph $(\{v_1, v_2, v_3, v_4\}, \{v_1v_2, v_1v_3, v_1v_4\})$ and add new nodes sequentially. For each new node, we connect it to two existing nodes. The probability of connecting a node w is proportional to the node degree of w. *Erdös-Rényi (ER) graph*. An Erdös-Rényi (ER) graph is a simple random graph



(a) AS1221 (b) AS1239 (c) AS1755 (d) AS3257

(e) AS3356 (f) AS3967 (g) AS6461 (h) AS7018

Fig. 9. Monitor assignment results of Scalpel. For each topology, we repeat each simulation 10 times and plot the results.

generated by connecting each pair of nodes with a fixed probability. In our simulations, this probability is set to be 0.015. *Random Geometric (RG) graph*. A Random Geometric (RG) graph is generated by first randomly deploying nodes in a unit square area and then connecting nodes when their distance is no larger than a threshold. In our simulations, we set the threshold to 0.08. *dK*-graph [23]. *dK*-graph models the degree distribution of a network and generate topologies with similar degree distribution. Then *dK*-graph generates network topologies with similar degree distribution but different scales. We use a real network topology as input and use *dK*-graph tool to generate larger graphs.

2) Autonomous system topologies. We also use real network topologies from the Rocketfuel [6] project to evaluate Scalpel. The topologies used are the autonomous system (AS) topologies, which represent IP-level connections between backbone/gateway routers of ASes from major Internet Service Providers (ISPs) (i.e., AT&T) around the world [2]. Each AS in Rocketfuel corresponds to an ISP.

B. Results

1) Effectiveness of Graph Trimming: Table I reports the graph trimming results of all topologies, including seven synthetic topologies and eight AS topologies. The first column is the topologies name. The second column and third column are the number of vertices and links in each topology. The other columns are the results in four different settings with different percentages of interesting links. $|\text{trim-}\mathcal{B}|$ denotes the number of bi-connected components which are trimmed in the first graph trimming stage. $|\text{trim-}\mathcal{T}|$ denotes the number of SPQR components which are trimmed in the second stage graph trimming. We also show the the number of bi-connected components in the original graph, as well as the number of SPQR components in the graph after the first stage graph trimming. For example, the 125.8/142 entry of the first row means that when 5% of the links are interesting links, there are 142 SPQR components (on average) after the first trimming stage and 125.8 (on average) of them are trimmed by the second trimming stage.

TABLE II. # OF MONITORS AND EXEC TIME COMPARED WITH MMP.

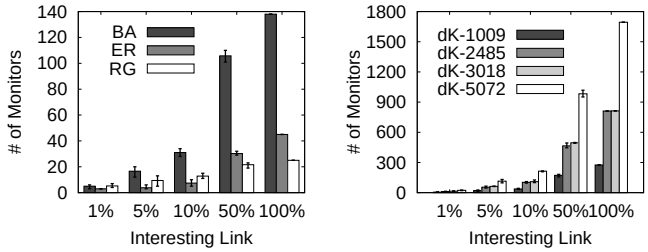
Network	1%		5%		10%		50%	
	|M|	time	|M|	time	|M|	time	|M|	time
AS1221	-0.96	+0.045	-0.88	+0.083	-0.82	+0.085	-0.34	+0.072
AS1239	-0.95	+0.033	-0.89	+0.06	-0.83	+0.047	-0.37	+0.078
AS1755	-0.89	+0.062	-0.77	+0.062	-0.69	+0.057	-0.29	+0.028
AS3257	-0.93	+0.09	-0.81	+0.054	-0.74	+0.05	-0.31	+0.045
AS3356	-0.9	+0.004	-0.82	+0.017	-0.75	+0.003	-0.29	+0.009
AS3967	-0.93	+0.041	-0.84	+0.037	-0.75	+0.053	-0.3	+0.045
AS6461	-0.94	+0.063	-0.87	+0.081	-0.8	+0.063	-0.39	+0.046
AS7018	-0.95	+0.045	-0.88	+0.051	-0.79	+0.047	-0.29	+0.053

From Table I, we first observe that when there are only a small number of interesting links (e.g., 1%), most of the bi-connected components and the SPQR components are trimmed by Scalpel. For example, when 1% links are interesting links in the *dK*-5072 topology, 1267.6 out of 1282 (on average) bi-connected components in the original graph are trimmed by the first trimming stage. Then, 381.3 out of 391 (on average) SPQR components in the graph are trimmed by the second trimming stage. Second, when the number of interesting links increases, the number of trimmed components decreases. There is also a special topology in the first row, which is a bi-connected component. In this case, the first trimming stage cannot trim any bi-connected component but the second trimming stage can still trim a large number of SPQR components.

2) Monitor Assignment: Then we show the monitor assignment results. Figure 9 shows the results in the eight AS topologies. The x-axis is the percentage of interesting links in all links, and the y-axis is the number of monitors assigned.

The blue horizontal line indicates the number of all vertices in each topology. We set the percentage of interesting links to 1%, 5%, 10%, 20%, ..., 100%. From Figure 9, we can see that when only 1% links are interesting links, only a small number of vertices are assigned as monitors. When more links are interesting links, more monitors should be assigned. Note that using MMP directly in the original graph to assign monitors is equivalent to the case when all links are interesting links. Therefore, using Scalpel assign monitors in the trimmed graph reduces the number of monitors than using MMP directly in the original graph. Table II shows the monitor reduction ratios in different networks with different settings. In the AS topologies, Scalpel is able to reduce up to 96% of the monitors compared with MMP. The table also shows that Scalpel needs similar execution time in various networks compared with MMP.

Figure 10(a) shows the number of monitors assigned in the three random topologies (i.e., BA, ER, RG) when there are different number of interesting links. As expected, less monitors are assigned when there are less interesting links. Although these random graphs have similar number of vertices (i.e., 300, 297, 296), they require different number of monitors. The main reason is that the BA graph has less links than the other two random graphs. Since a dense network can enable more measurement paths to identify interesting links, it usually requires less monitors than a sparse network. Figure 10(b) shows the number of monitors assigned in the four topologies generated by the *dK*-graph tool [23]. When there are more interesting links, more monitors are required to be assigned.



(a) random topologies (b) *dK*-graph topologies

Fig. 10. Monitor assignment results of synthetic topologies.

VIII. Conclusion

In this paper, we propose Scalpel, an efficient preferential link tomography approach. We theoretically prove that the graph trimming algorithm in Scalpel is minimal and the obtained assignment by Scalpel is able to identify all interesting links in the original graph. Extensive simulations based on both synthetic topologies and real network topologies show the effectiveness of Scalpel. Compared with state-of-the-art, our approach reduces the number of monitors by 39.0%$\sim$98.6% when 50%$\sim$1% of all links are interesting links.

There are multiple dimensions to explore in the future work. First, we would like to investigate how to design an optimal monitor assignment algorithm so that the number of assigned monitors can be minimized. It is possible since our graph trimming algorithm guarantees that an optimal monitor assignment in the graph after trimming is also an optimal monitor assignment in the original graph. Second, we would like to investigate how to select measurement paths to facilitate the identification of link metrics.

References

[1] Y. Vardi, "Network Tomography: Estimating Source-Destination Traffic Intensities from Link Data," *Journal of the American Statistical Association*, vol. 91, no. 433, pp. 365–377, 1996.

[2] 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. of ACM IMC*, 2013.

[3] E. Lawrence, G. Michailidis, V. N. Nair, and B. Xi, "Network Tomography: A Review and Recent Developments," in *Frontiers in Statistics*, 2006.

[4] R. Kumar and J. Kaur, "Practical Beacon Placement for Link Monitoring Using Network Tomography," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 2196–2209, 2006.

[5] 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, 2012.

[6] N. T. Spring, R. Mahajan, D. Wetherall, and T. E. Anderson, "Measuring ISP Topologies with Rocketfuel," *IEEE/ACM Trans. Netw.*, vol. 12, no. 1, pp. 2–16, 2004.

[7] L. Mo, Y. He, Y. Liu, J. Zhao, S. Tang, X. Li, and G. Dai, "Canopy Closure Estimates with GreenOrbs: Sustainable Sensing in the Forest," in *Proceedings of SenSys*, 2009.

[8] X. Mao, X. Miao, Y. He, T. Zhu, J. Wang, W. Dong, X. yang Li, and Y. Liu, "Citysee: Urban CO2 Monitoring with Sensors," in *Proceedings of INFOCOM*, 2012.

[9] W. Dong, Y. Liu, Y. He, and T. Zhu, "Measurement and Analysis on the Packet Delivery Performance in A Large Scale Sensor Network," in *Proceedings of INFOCOM*, 2013.

[10] A. B. Downey, "Using Pathchar to Estimate Internet Link Characteristics," in *in Proc. of ACM SIGCOMM*, 1999.

[11] G. Jin, G. Yang, B. R. Crowley, and D. A. Agarwal, "Network Characterization Service (NCS)," in *in Proc. of IEEE HPDC*, 2001.

[12] Y. Bejerano and R. Rastogi, "Robust Monitoring of Link Delays and Faults in IP Networks," *IEEE/ACM Trans. Netw.*, vol. 14, no. 5, pp. 1092–1103, 2006.

[13] J. D. Horton and A. Lpez-Ortiz, "On the Number of Distributed Measurement Points for Network Tomography," in *in Proc. of ACM IMC*, 2003.

[14] H. H. Song, L. Qiu, and Y. Zhang, "Netquest: A flexible framework for large-scale network measurement," *IEEE/ACM Trans. Netw.*, vol. 17, no. 1, pp. 106–119, 2009.

[15] Y. Chen, D. Bindel, H. Song, and R. H. Katz, "An algebraic approach to practical and scalable overlay network monitoring," in *in Proc. of ACM SIGCOMM*, 2004.

[16] O. Gurewitz and M. Sidi, "Estimating One-way Delays from Cyclic-Path Delay Measurements," in *in Proc. of IEEE INFOCOM*, 2001.

[17] S. Ahuja, S. Ramasubramanian, and M. Krunz, "SRLG Failure Localization in All-Optical Networks Using Monitoring Cycles and Paths," in *in Proc. of IEEE INFOCOM*, 2008.

[18] L. Ma, T. He, K. K. Leung, D. Towsley, and A. Swami, "Efficient Identification of Additive Link Metrics via Network Tomography," in *in Proc. of IEEE ICDCS*, 2013.

[19] R. E. Tarjan, "Depth-First Search and Linear Graph Algorithms." *SIAM J. Comput.*, vol. 1, no. 2, pp. 146–160, 1972.

[20] J. E. Hopcroft and R. E. Tarjan, "Dividing a Graph into Triconnected Components." *SIAM J. Comput.*, vol. 2, no. 3, pp. 135–158, 1973.

[21] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Monitor Placement for Maximal Identifiability in Network Tomography," in *in Proc. of IEEE INFOCOM*, 2014.

[22] Y. Gao, D. W. Wu, Wenbin, C. Chen, X.-Y. Li, and J. Bu, "Preferential link tomography: Monitor assignment for inferring interesting link metrics," Tech. Rep. [Online]. Available: http://www.emnets.org/pub/gaoyi/scalpel-tech.pdf

[23] P. Mahadevan, C. Hubble, D. V. Krioukov, B. Huffaker, and A. Vahdat, "Orbis: Rescaling Degree Correlations to Generate Annotated Internet Topologies," in *in Proc. of ACM SIGCOMM*, 2007.