Optimal Monitor Assignment for Preferential Link Tomography in Communication Networks

Wei Dong, *Member, IEEE*, Yi Gao, *Member, IEEE*, *ACM*, Wenbin Wu, Jiajun Bu, *Member, IEEE*, *ACM*, Chun Chen, *Member, IEEE*, and Xiang-Yang Li, *Fellow, IEEE*

***Abstract*— Inferring fine-grained link metrics by using aggregated path measurements, known as network tomography, is an effective and efficient way to facilitate various network operations, such as network monitoring, load balancing, and failure diagnosis. Given the network topology and a set of interesting links, we study the problem of calculating the link metrics of these links by end-to-end cycle-free path measurements among selected monitors, i.e., preferential link tomography. Since assigning nodes as monitors usually requires non-negligible operational cost, we focus on assigning a minimum number of monitors to identify these interesting links. We propose an optimal monitor assignment (OMA) algorithm for preferential link tomography in communication networks. OMA first partitions the graph representing the network topology into multiple graph components. Then, OMA carefully assigns monitors inside each graph component and at the boundaries of multiple graph components. We theoretically prove the optimality of OMA by proving: 1) the monitors assigned by OMA are able to identify all interesting links and 2) the number of monitors assigned by OMA is minimal. We also implement OMA and evaluate it through extensive simulations based on both real topologies and synthetic topologies. Compared with two baseline approaches, OMA reduces the number of monitors assigned significantly in various network settings.**

***Index Terms*— Network tomography, optimal monitor assignment, network measurement.**

I. Introduction

UNDERSTANDING of the internal behaviors of a large scale network is essential for various operations [1], [2], such as network monitoring, load balancing, and failure diagnosis. Directly measuring the internal network behaviors (e.g., delays on individual links), however, is not always feasible due to the traffic overhead and the lack of support at internal network elements for making such measurements [3]. *Network tomography* techniques [4], instead, focuses on using end-to-end measurements to infer hop-by-hop link metrics, providing an effective and efficient way to improve the network visibility. Specifically, a number of nodes with monitoring capabilities, i.e., *monitors*, are able to initiate/collect end-to-end measurements of selected cycle-free paths. Then, by using these measurements, the internal link metrics can be decomposed after solving a system of equations.

In many cases, these link metrics are additive [3], [5]. For example, delay is a typical additive metric, 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. In the linear system, the unknown variables are the link metrics, and the known constants are the end-to-end path measurements, each equals to the sum of the corresponding link metrics along the routing path [3]. Since these end-to-end path measurements are conducted between pairs of monitors, assigning which nodes to be monitors (we refer to both the monitor assigning problem and the monitors assigned as *monitor assignment*) 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, v is a node with degree (the number of links that incident to a node) two and l_1, l_2 are the only two links that incident to v. When node v is not assigned as a monitor, no matter how to conduct cycle-free measurement paths among other monitors, we can only get the sum of the link metrics of l_1, l_2, instead of the link metric of each link. On the other hand, we want to minimize the number of monitors assigned, because assigning nodes as monitors usually needs non-negligible operational cost (e.g., hardware/software, human efforts).

In communication networks, some links, such as problematic links reported by customers or links located in critical infrastructures (e.g., hospitals and fire departments), are known to be more important than other links. Therefore, given the network topology of a network and a set of interesting links (i.e., preferential links), how to assign a minimum number of monitors to identify these interesting links, i.e., the optimal monitor assignment, is the key problem studied in this paper. Finding the optimal monitor assignment is a challenging task due to the following two reasons. First, it is difficult to

Manuscript received February 26, 2015; revised October 24, 2015 and February 24, 2016; accepted May 18, 2016; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor J. Wang. Date of publication June 30, 2016; date of current version February 14, 2017. This work was supported in part by the National Science Foundation of China under Grants 61502417 and 61472360, the Zhejiang Provincial Natural Science Foundation of China under Grant LY16F020006, the National Key Technology Research and Development Program under Grant 2014BAK15B02, the Fundamental Research Funds for the Central Universities under Grants 2016FZA5010 and 2016QNA5012, and the China Ministry of Education–China Mobile Joint Project under Grant MCM20150401. The work of X.-Y. Li was supported in part by the National Science Foundation of China under Grants ECCS-1247944, ECCS-1343306, CMMI 1436786, and CNS 1526638 and the National Natural Science Foundation of China under Grant 61520106007. *(Corresponding author: Yi Gao.)*

W. Dong, Y. Gao, W. Wu, J. Bu, and C. Chen are with the College of Computer Science, Zhejiang University, Zhejiang 310027, China (e-mail: dongw@zju.edu.cn; gaoyi@zju.edu.cn; wuwb@zju.edu.cn; bjj@zju.edu.cn; chenc@zju.edu.cn).

X.-Y. Li is with the School of Computer Science and Technology, University of Science and Technology of China, Hefei 230026, China (e-mail: xiangyangli@ustc.edu.cn).

Digital Object Identifier 10.1109/TNET.2016.2581176

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 that no interesting link depends on. Second, measurement paths between monitors can cross boundaries of different graph components (e.g., bi-connected [6]/tri-connected [7] components), causing the monitor assignment at these boundaries to depend on multiple graph components. As a result, assigning monitors for each graph component separately is not sufficient.

In this paper, we propose OMA, an *Optimal* Monitor Assignment algorithm for the preferential link tomography problem. Given a graph $\mathcal{G}$ representing a network and a set of interesting links, we first partitions the graph $\mathcal{G}$ into a number of graph components [6], [7] and use a graph trimming algorithm to trim some unrelated graph components [8]. Then OMA takes the output of the graph trimming, i.e., a trimmed graph $\mathcal{G}_t$ and a set $\mathcal{H}$ of *helper vertices* [8], as input and assigns monitors. The output of OMA is an optimal monitor assignment to identify the interesting links of the original graph $\mathcal{G}$. For each graph component, OMA carefully assigns vertices in the trimmed graph $\mathcal{G}_t$ or in the helper vertex set $\mathcal{H}$ as monitors, to achieve the optimal monitor assignment. Further, in order to assign minimum monitors at the boundaries of multiple graph components, OMA includes a *Chain-rule* formulation and solves it by a novel algorithm.

We theoretically prove the optimality of OMA by proving 1) the monitors assigned by OMA are able to identify all interesting links in the original graph, and 2) the number of monitors assigned by OMA is minimal. We also implement OMA and evaluate it through extensive simulations based on both real network topologies and synthetic topologies. The time complexity of OMA 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 OMA significantly reduces the number of monitors assigned in various networks, compared with the two baseline approaches.

The contributions of this paper are summarized as follows.

- We propose OMA, an optimal monitor assignment algorithm for preferential link tomography in communication networks.
- We theoretically prove the optimality of OMA, which says the number of monitors assigned by OMA is minimal for identifying all interesting links in the original graph.
- We implement and evaluate OMA by extensive simulations. The time complexity of OMA 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 OMA reduces the number of monitors significantly.

The rest of the paper is organized as follows. Section II discusses the related work. Section III gives the network model and some background information about monitor assignment. Section IV describes the optimal monitor assignment algorithm in detail. Section V theoretically proves the optimality of OMA. Section VI presents the evaluation of OMA. and finally, Section VII concludes this paper.

II. Related Work

Knowledge of the internal characteristics (e.g., link delays) of a network is essential for network monitoring, failure diagnosis, load balancing and other network operations. In order to measure network metrics, different approaches have been proposed. The first category includes *hop-by-hop* approaches, which use diagnostic tools such as *traceroute*, *pathchar* [9], and *Network Characterization Service (NCS)* [10] to measure hop-by-hop link metrics directly. *Traceroute* measures the hop-by-hop delay by sending multiple probes with different time-to-live (TTL) fields. *Pathchar* also uses probes to measure hop-by-hop delays, capacities and loss rates. *NCS* reports the available capacity of each hop. These 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 calculate hop-by-hop link metrics. When the link metrics are modeled as random variables with certain distributions, many multicast-based methods are used to perform network tomography [11], [12]. In [1] and [13], the necessary and sufficient conditions on the multicast tree are studied for identifying all links. In [14], the Fourier transform of the observable distributions to calculate the unobservable distributions. He *et al.* [15] propose a probe allocation framework to optimize the link metric estimation accuracy. It provides a systematic solution to the noisy measurement problem. These approaches all assume that the network is identifiable. In contrast, we focus on the problem of optimal monitor assignment to make interesting links identifiable.

For constant link metrics [1]–[3], [16]–[18], many approaches use the path measurements to calculate link metrics, reducing the number of monitors and probes significantly. The basic idea is to build a linear system from the path measurements and use linear algebra to calculate the unknown link metrics [19], [20]. As shown in [19], the problem is challenging since many path measurements are linearly dependent. Several approaches [3], [20], [21] 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 requirement of the network topology to identify all failed links. Many approaches focus on the general case when the link metrics are arbitrary valued. Gopalan and Ramasubramanian [5], [22] first analyze the number of linearly independent cycles/paths, then give the necessary and sufficient conditions on the network topology to use cyclic measurement paths for identifying link metrics. Since routing along cycles is typically prohibited in real networks, cycle-free measurement path are usually preferred.

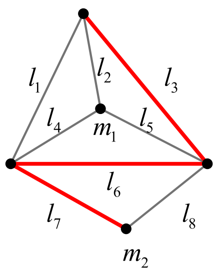
Ma *et al.* [3] give the necessary and sufficient conditions on the network topology when only cycle-free measurement paths are allowed. In order to identify all links in a network, Ma et al. also propose an efficient algorithm MMP [3] to assign the minimum number of monitors as well as an efficient path construction algorithm [23]. Further, a robust network tomography approach [24] is also proposed to select measurement paths intelligently for better performance in the presence of network element failures. Given the number of monitors, a near optimal monitor assignment algorithm [25] is proposed to achieve maximum identifiability. Different with these approaches, Scalpel [8] studies a more general case when we are only interested in a subset of links. Scalpel first trims the original graph and then reuses MMP to assign monitors. Compared with MMP, Scalpel reduces the number of monitors significantly when there are only a small number of interesting links. However, Scalpel is still not optimal in terms of the number of monitors assigned. In some cases, Scalpel will assign much more monitors than necessary. Different with Scalpel, OMA is an optimal monitor assignment for the preferential link tomography problem.

III. Network Model and Background on Monitor Assignment

A. Network Model and Assumptions

We assume that the network topology is known and does not change during the measurement process. 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. Since different connected components of the network can be monitored separately, we can assume that graph $\mathcal{G}$ is connected. We denote the link that incidents to vertices u, v by uv. We assume that the links are symmetric, i.e, the link metrics of uv and vu are the same. When links are asymmetric, it has been proven [26] that a link is unidentifiable unless its two endpoints are both monitors. We also 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 whose link metrics are interested to be identified. A subset of vertices in $V(\mathcal{G})$ are assigned as monitors and can initiate/collect end-to-end measurements for identifying the links in $\mathcal{I}$. Since routing along cycles is typically prohibited in real networks [3], we only use cycle-free measurement paths between two monitors.

We study the network tomography problem in the context of *controllable* measurements, i.e., monitors can control the routing paths of the measurement packets. There are several reasons for this controllable measurements assumption. First, such routing is generally supported in common networks like overlay networks and single-ISP networks [3]. In these networks, the controllable measurements can be achieved by enabling source routing (rfc0791 [27]) at the routers/switches. Second, it has been shown that controllable routing is able to lower the network latency and increase the available bandwidth, by fully or partially specifying the routing paths of packets [28]–[30]. Finally, a recent study [31] shows that source routing can help the network manager address problems like convergence and controller placement in networks performing Software-Defined Networking (SDN). In SDN, a (logically) centralized SDN controller can dictate paths of measurement packets by using control plane messages to set the forwarding table entries of switches in the network. Recently, Hu et al. propose XPath [32], which is a simple, practical and readily-deployable way to implement controllable routing, using existing commodity switches. In XPath, a node first queries the controller for a path ID to its intended destination, then sends the packet according to the path specified by the path ID. Therefore, XPath can be directly used to implement controllable measurement.



$$m_1 \rightarrow m_2: \quad \begin{cases} l_5 + l_3 + l_1 + l_7 = W_1 \\ l_5 + l_6 + l_7 = W_2 \\ l_2 + l_3 + l_6 + l_7 = W_3 \\ l_2 + l_1 + l_7 = W_4 \\ l_4 + l_7 = W_5 \\ l_4 + l_6 + l_8 = W_6 \\ l_5 + l_8 = W_7 \end{cases} \Rightarrow \begin{cases} l_3 = \dfrac{W_1 + W_3 - W_2 - W_4}{2} \\ l_6 = \dfrac{W_6 + W_2 - W_5 - W_7}{2} \end{cases}$$

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, no matter how to conduct cycle free measurement paths between these two monitors.

A monitor m_A can initiate a measurement packet to another monitor m_B. Then monitor m_B can obtain the path measurement, which is the sum of all link metrics along the path. Then we can build a linear system to identify the interesting links based on these path measurements. If a link metric can be calculated from the linear system obtained by a monitor assignment, this link is *identifiable* by the monitor assignment.

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.

B. Concepts and Definitions

The following concepts in graph theory 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. Note that partitioning a connected graph into bi-connected components is a classical graph theory problem which can be solved in linear time [6].
- 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 (including a triangle). Note that there exists a classical graph theory algorithm [7] called SPQR-tree[1] which can partition a bi-connected graph into a number

[1]There are four possible components in a typical SPQR-tree [7]. 1) S node, which is a cycle (i.e., polygon, we will give more discussion about it after showing an example in Figure 2). 2) P node, which has multiple links between two vertices. In this paper, we assume that two vertices can only have at most one link. Therefore, there is no P node after graph partition. 3) Q node, which is just a single link. In this paper, a single is considered as a special case. 4) R node, which is a tri-connected component.

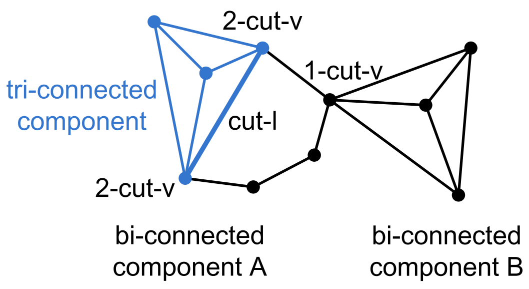


Fig. 2. A sample graph to illustrate some graph theory concepts used in this paper. The 1-cut-v separates the graph into two bi-connected components. The two 2-cut-vs separate one bi-connected component into one tri-connected component and a cycle. The link cut-l connects the two 2-cut-vs.

SPQR components (tri-connected components and/or cycles in this paper) 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 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 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 cycle by a pair of 2-cut-vs. The link that connects to the two 2-cut-vs is a cut-l.

It is worth noting that the SPQR-tree algorithm will add some *virtual links* to the components after graph partitioning. In the example shown in Figure 2, if the cut-l does not exist in the original graph, the SPQR-tree algorithm will add this link to the tri-conneected component and the cycle, as a virtual link. It has been proved in [25] that the virtual links do not affect the link identifiability of tri-connected components (except triangles). For cycles, when at least one of its neighboring SPQR components is a tri-connected component (not triangle), whether the cut-l between them is a virtual link does not affect the link identifiability. For cycles (including triangles) share one virtual link, the virtual link will affect the identifiability of these cycles (or triangles). In order to avoid this, we modify the SPQR-tree algorithm so that these cycles (or triangles) are merged to a larger cycle. As a result, the virtual link will not affect the link identifiability. In the rest of the paper, we will view the virtual links as normal links.

We then give the definitions of a *monitor assignment* and an *optimal monitor assignment* formally.

Definition 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 $\mathcal{G}_t$.

Since we want to assign a minimum number of monitors, we have the following definition about optimal monitor assignment.

*Definition 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 **minimal** for identifying all the interesting links in the **original graph** $\mathcal{G}$.*

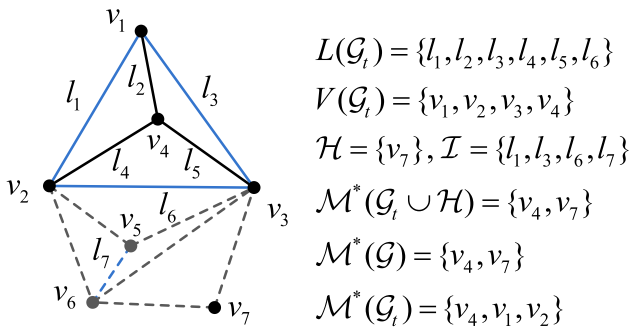

v_1
l_1
l_2
l_3
v_4
l_4
l_5
v_2
v_3
l_6
v_5
l_7
v_6
v_7
$L(\mathcal{G}_t) = \{l_1, l_2, l_3, l_4, l_5, l_6\}$
$V(\mathcal{G}_t) = \{v_1, v_2, v_3, v_4\}$
$\mathcal{H} = \{v_7\}, \mathcal{I} = \{l_1, l_3, l_6, l_7\}$
$\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H}) = \{v_4, v_7\}$
$\mathcal{M}^*(\mathcal{G}) = \{v_4, v_7\}$
$\mathcal{M}^*(\mathcal{G}_t) = \{v_4, v_1, v_2\}$

Fig. 3. An example that illustrates the output of graph trimming [8], which is a trimmed graph $\mathcal{G}_t$ (solid part in this example) and a set $\mathcal{H}$ of helper vertices. 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.

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 $\mathcal{G}_x$.

C. Background on Graph Trimming

Since the input of OMA is the output of the graph trimming algorithm [8], we give some related background about this graph trimming algorithm in this subsection.

Given an original graph $\mathcal{G}$ and a set $\mathcal{I}$ of interesting links, the graph trimming algorithm trims the graph by two stages. The first stage trims a number of bi-connected components and the second stage trims a number of SPQR components (i.e., tri-connected components or cycles). During the second stage graph trimming, a helper vertex is chosen for each SPQR component been trimmed. Helper vertices of multiple adjacent SPQR components been trimmed will be merged into one helper vertex. Then each helper vertex is associated with a cut-l (e.g., cut-l in Figure 2) which separates the trimmed graph and a number of adjacent SPQR components been trimmed iteratively. For a cut-l l and its helper vertex v, we use $v = h(l)$ and $l = h^{-1}(v)$ to denote this association. After graph trimming, the output is a trimmed graph $\mathcal{G}_t$ and a set $\mathcal{H}$ of helper vertices. The reason of reserving the helper vertices is that sometimes assigning one helper vertex as a monitor can achieve the same identifiability as assigning two monitors in $\mathcal{G}_t$. Therefore, reserving these helper vertices keeps the possibility of finding an optimal solution after graph trimming.

Figure 3 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. The link that associates with v_7 is $l_6 = h^{-1}(v_7)$. According to Definition 1 and Definition 2, we can use the following notation to define different monitor assignments.

- $\mathcal{M}(\mathcal{G})$ is a monitor assignment in the original graph $\mathcal{G}$. Note that this assignment may not be able to identify all interesting links.
- $\mathcal{M}^*(\mathcal{G})$ is an optimal monitor assignment in the original graph $\mathcal{G}$, which is the optimal solution we want to obtain. In this example, $\{v_4, v_7\}$ is an optimal solution.
- $\mathcal{M}(\mathcal{G}_t \cup \mathcal{H})$ is a monitor assignment in $V(\mathcal{G}_t) \cup \mathcal{H}$.
- $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ is an optimal monitor assignment in $V(\mathcal{G}_t) \cup \mathcal{H}$. According to Theorem VI.2 [8], a $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ is also a $\mathcal{M}^*(\mathcal{G})$. Therefore, OMA takes $\mathcal{G}_t$ and $\mathcal{H}$

Algorithm 1 OMA

```
Input: The trimmed graph $\mathcal{G}_t$, the helper vertex set $\mathcal{H}$ and the
    interesting link set $\mathcal{I} \neq \emptyset$
Output: A monitor assignment $\mathcal{M}^S(\mathcal{G}_t \cup \mathcal{H})$ ($\mathcal{M}^S$) of vertices
    in $V(\mathcal{G}_t \cup \mathcal{H})$ as monitors
1: let $\mathcal{M}^S$ be a monitor assignment without any monitors
2: let $\mathcal{B}_1, \mathcal{B}_2$... be the bi-connected components partitioned
3: for each bi-connected component $\mathcal{B}_i$ do
4:     let $\mathcal{T}_1, \mathcal{T}_2$... be SPQR components partitioned
5:     for each tri-connected component $\mathcal{T}_j$ do
6:         ASSIGN-TRI($\mathcal{T}_j, \mathcal{H}, \mathcal{I}, \mathcal{M}^S$)
7:     for each cycle $\mathcal{T}_j$ do
8:         ASSIGN-CYCLE($\mathcal{T}_j, \mathcal{H}, \mathcal{I}, \mathcal{M}^S$)
9: ASSIGN-BI($\mathcal{G}_t, \mathcal{H}, \mathcal{I}, \mathcal{M}^S$)
```

as input and aims at finding the optimal solution in $V(\mathcal{G}_t) \cup \mathcal{H}$.

- $\mathcal{M}^*(\mathcal{G}_t)$ is an optimal monitor assignment in the trimmed graph $\mathcal{G}_t$. The difference of this assignment and $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ is that it cannot assign vertices in $\mathcal{H}$ as monitors. As a result, $\mathcal{M}^*(\mathcal{G}_t)$ may have a larger number of monitors compared with $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$. In the example shown in Figure 3, $\{v_4, v_1, v_2\}$ is an optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_t)$ and $\{v_4, v_7\}$ is an optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$. We can see that reserving a number of helper vertices is able to keep the possibility of finding the optimal solution.

IV. Optimal Monitor Assignment

A. Overview

Algorithm 1 gives the framework of assigning monitors in $V(\mathcal{G}_t) \cup \mathcal{H}$. The input is the trimmed graph $\mathcal{G}_t$, the helper vertex set $\mathcal{H}$ and the interesting link set $\mathcal{I}$. The output is a monitor assignment $\mathcal{M}^S(V(\mathcal{G}_t) \cup \mathcal{H})$ (or $\mathcal{M}^S$ for short) of vertices in $V(\mathcal{G}_t \cup \mathcal{H})$. After initializing the assignment $\mathcal{M}^S$ (line 1), OMA assigns monitors for each tri-connected component or cycle (line 2 to 8) by two different algorithms (line 6 and 8). After that, there are still some boundary cases about the bi-connected components need to be handled with (e.g., the $\mathcal{G}_t$ is just a single interesting link). Therefore, an algorithm Assign-Bi(.) is used to handle these boundary cases.

In the monitor assignment process, there are two fundamental problems to be addressed. 1) *How to assign monitors in each graph component?* As mentioned in the introduction section, the identifiability of a certain monitor assignment does not only depend on a single graph component, but depends on multiple graph components. However, we found that within each graph component, some monitors must be assigned no matter how to assign monitors outside that graph component. Based on this finding, OMA can assign some monitors within each graph component without sacrificing the optimality (Section IV.B). 2) *How to assign monitors at the boundaries of multiple graph components?* Although OMA can assign a number of monitors within each graph component, it still needs to assign monitors at the boundaries of multiple components. In order to address the monitor assignment problem with multiple components dependency, we propose a novel *Chain-rules* formulation to model the dependency and an efficient solver to assign a minimum number of monitors (Section IV.C).

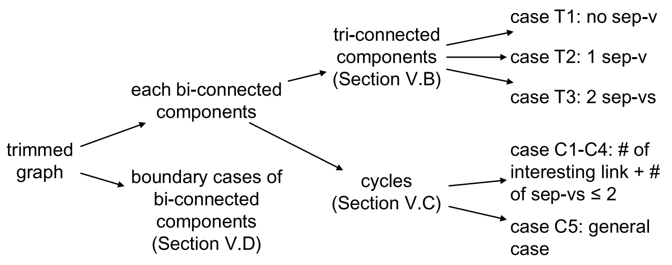

trimmed graph
each bi-connected components
boundary cases of bi-connected components (Section V.D)
tri-connected components (Section V.B)
cycles (Section V.C)
case T1: no sep-v
case T2: 1 sep-v
case T3: 2 sep-vs
case C1-C4: # of interesting link + # of sep-vs ≤ 2
case C5: general case

Fig. 4. Overview of the OMA algorithm.

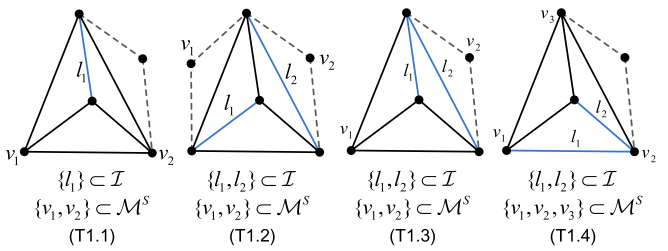

l_1
v_1
v_2
$\{l_1\} \subset \mathcal{I}$
$\{v_1, v_2\} \subset \mathcal{M}^S$
(T1.1)
l_2
$\{l_1, l_2\} \subset \mathcal{I}$
$\{v_1, v_2\} \subset \mathcal{M}^S$
(T1.2)
$\{l_1, l_2\} \subset \mathcal{I}$
$\{v_1, v_2\} \subset \mathcal{M}^S$
(T1.3)
v_3
$\{l_1, l_2\} \subset \mathcal{I}$
$\{v_1, v_2, v_3\} \subset \mathcal{M}^S$
(T1.4)

Fig. 5. An example of the first case of assigning monitors in a tri-connected component.

In the following, we will describe the monitor assignment process in detail. Figure 4 gives an overview of this process. We will start from assigning monitors in each tri-connected component (Section IV.B). Three different cases with different number of sep-vs will be discussed. Then we will describe the monitor assignment in each cycles (Section IV.C). A general case will be discussed in detail after four special cases. Finally, we will describe the boundary cases of multiple bi-connected components ((Section IV.D)).

B. Monitor Assignment for Tri-Connected Components

Algorithm 2 assigns monitors for each tri-connected component in $\mathcal{G}_t$. It takes $\mathcal{T}_j, \mathcal{H}, \mathcal{I}, \mathcal{M}^S$ as input and updates the monitor assignment $\mathcal{M}^S$ when necessary. For each tri-connected component $\mathcal{T}_j$, the monitor assignment inside it highly depends on the number of sep-vs it has. For example, if $\mathcal{T}_j$ has no sep-v, it means that the trimmed graph $\mathcal{G}_t$ is just a single tri-connected graph. In this case, we do not need to consider measurement path from/to other components of $\mathcal{T}_j$. Therefore, in the following, we consider three different cases (line 2, 16 and 30) according to how many sep-vs $\mathcal{T}_j$ has.

The first case **T1** (line 2 to 15) is that $\mathcal{T}_j$ has no sep-v, which means the trimmed graph $\mathcal{G}_t$ is a tri-connected graph. If there is only one interesting link l, OMA assigns the two endpoints of l as monitors. If not, we define two vertex set V_1, V_2. In V_1, each vertex is in $\mathcal{T}_j$ and is not in any interesting link in $\mathcal{T}_j$. In V_2, each vertex is a helper vertex of a link in $\mathcal{T}_j$ and does not in any interesting link. Then OMA assigns two or three vertices as monitors in four difference cases. **T1.1** If V_1 has at least two vertices, OMA assigns any two vertices in V_1 as monitors. Figure 5(T1.1) is an example of this case where v_1, v_2 are assigned as two monitors.

Algorithm 2 Minimum Monitor Assignment: Assign-Tri

Input: Tri-connected component $\mathcal{T}_j$, the helper vertex set $\mathcal{H}$, the interesting link set $\mathcal{I}$, the monitor assignment $\mathcal{M}^S$

Output: An updated monitor assignment $\mathcal{M}^S$

```
1:  procedure ASSIGN-TRI($\mathcal{T}_j, \mathcal{H}, \mathcal{I}, \mathcal{M}^S$)
2:      if $\mathcal{T}_j$ has no sep-v then
3:          if $|L(\mathcal{T}_j) \cap \mathcal{I}| = \{l\}$ then
4:              $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup V(l)$
5:              return
6:          $V_1 = \{v | L(v) \cap \mathcal{I} = \emptyset, v \in V(\mathcal{T}_j)\}$
7:          $V_2 = \{v | h^{-1}(v) \in L(\mathcal{T}_j), v \cap V(\mathcal{I}) = \emptyset\}$
8:          if $|V_1| \geq 2$ then
9:              $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v_1, v_2\}$, $\{v_1, v_2\} \subseteq V_1$
10:         else if $|V_2| \geq 2$ then
11:             $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v_1, v_2\}$, $\{v_1, v_2\} \subseteq V_2$
12:         else if $V_1 = \{v_1\}, V_2 = \{v_2\}, v_1 \notin V(h^{-1}(v_2))$ then
13:             $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v_1, v_2\}$
14:         else
15:             $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v_1, v_2, v_3\}$, $\{v_1, v_2, v_3\} \subseteq V(\mathcal{T}_j)$
16:     else if $\mathcal{T}_j$ has one sep-v $s$ then
17:         $L_s = L(\mathcal{T}_j) \cap L(s) \cap \mathcal{I}$
18:         if $|L_s| = 0$ then
19:             if $\exists v \notin V(\mathcal{I}), h^{-1}(v) \in L(\mathcal{T}_j), s \notin V(h^{-1}(v))$ then
20:                 $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v\}$
21:             else if $\exists v \in (V(\mathcal{T}_j) - s), v \notin V(\mathcal{I})$ then
22:                 $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v\}$
23:             else
24:                 $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v_1, v_2\}$, $\{v_1, v_2\} \subseteq V(\mathcal{T}_j) - s$
25:         else if $|L_s| = 1$ then
26:             $v = V(L_s) - s$
27:             $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v\}$
28:         else
29:             $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v_1, v_2\}$, $\{v_1, v_2\} \subseteq V(\mathcal{T}_j) - s$
30:     else if $\mathcal{T}_j$ has two sep-v $s_1, s_2$ then
31:         if $\exists v \in (V(\mathcal{T}) - \{s_1, s_2\}), v \notin V(\mathcal{I})$ then
32:             $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v\}$
33:         else if $\exists v, h^{-1}(v) \in L(\mathcal{T}_j) - \{s_1 s_2\}, v \notin V(\mathcal{I})$ then
34:             $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v\}$
35:         else
36:             $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v\}$, $\{v\} \subseteq V(\mathcal{T}_j) - \{s_1, s_2\}$
```

T1.2 If V_2 has at least two vertices, OMA assigns any two vertices in V_2 as monitors. Figure 5(T1.2) is an example of this case. **T1.3** If V_1, V_2 both have one vertex (v_1, v_2), and v_1 is not in the link associated with v_2, OMA assigns these two vertices v_1, v_2 as monitors. Figure 5(T1.3) is an example of this case. **T1.4** Otherwise, OMA assigns any three vertices in $\mathcal{T}_j$ as monitors. Figure 5(T1.4) is an example of this case, where v_1, v_2, v_3 are assigned as three monitors.

The second case **T2** (line 16 to 29) is that $\mathcal{T}_j$ has only one sep-v s. We first define a link set L_s as all interesting links that in $\mathcal{T}_j$ and incident to s. Then OMA assigns one or two monitors in three difference cases. **T2.1** If L_s does not include any link, OMA will first try to assign a helper vertex v as a monitor (**T2.1.1**), when v does not in any interesting links, the link $h^{-1}(v)$ associated with v is in $L(\mathcal{T}_j)$ and s is not in $h^{-1}(v)$. If no such vertex (**T2.1.2**), the second choice of OMA is to assign a vertex in $V(\mathcal{T}_j) - s$ which does not in any interesting link as a monitor. Figure 6(T2.1.1 or T2.1.2) shows this case. OMA will try to assign v_1 or v_2 as a monitor. Otherwise (**T2.1.3**), assigning one vertex as monitor cannot identify all interesting link in $\mathcal{T}_j$. Therefore, OMA assigns two vertices in $V(\mathcal{T}_j) - s$ as monitors. In the example shown in Figure 6(T2.1.3), OMA will assign v_1, v_2 as two monitors. **T2.2** If L_s has a single link, let vs be this link. Then OMA will assign v as a monitor. In Figure 6(T2.2), OMA will assign v_1 as a monitor. **T2.3** Otherwise, OMA assigns any two vertices in $\mathcal{T}_j - s$ as monitors. In Figure 6(T2.3), OMA can assign v_1, v_2 as two monitors.

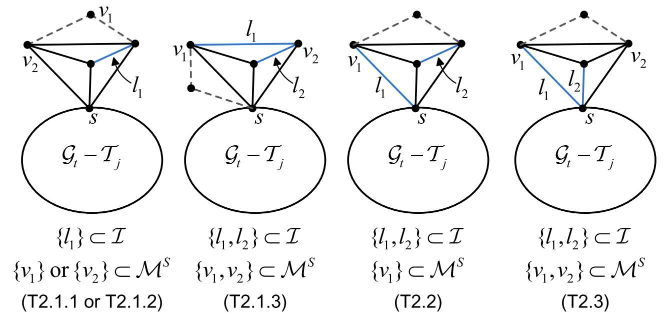

v_1
v_2
l_1
s
$\mathcal{G}_t - \mathcal{T}_j$
$\{l_1\} \subset \mathcal{I}$
$\{v_1\}$ or $\{v_2\} \subset \mathcal{M}^S$
(T2.1.1 or T2.1.2)
l_1
v_1
v_2
l_2
s
$\mathcal{G}_t - \mathcal{T}_j$
$\{l_1, l_2\} \subset \mathcal{I}$
$\{v_1, v_2\} \subset \mathcal{M}^S$
(T2.1.3)
v_1
l_1
l_2
s
$\mathcal{G}_t - \mathcal{T}_j$
$\{l_1, l_2\} \subset \mathcal{I}$
$\{v_1\} \subset \mathcal{M}^S$
(T2.2)
v_1
v_2
l_1
l_2
s
$\mathcal{G}_t - \mathcal{T}_j$
$\{l_1, l_2\} \subset \mathcal{I}$
$\{v_1, v_2\} \subset \mathcal{M}^S$
(T2.3)

Fig. 6. An example of the second case of assigning monitors in a tri-connected component.

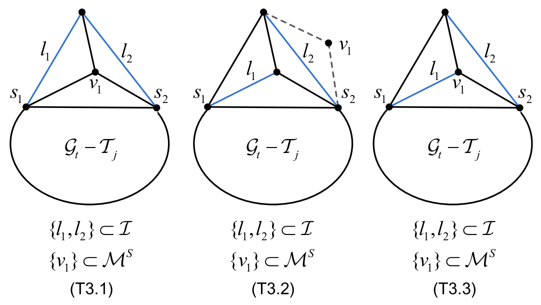

l_1
l_2
s_1
v_1
s_2
$\mathcal{G}_t - \mathcal{T}_j$
$\{l_1, l_2\} \subset \mathcal{I}$
$\{v_1\} \subset \mathcal{M}^S$
(T3.1)
l_1
l_2
v_1
s_1
s_2
$\mathcal{G}_t - \mathcal{T}_j$
$\{l_1, l_2\} \subset \mathcal{I}$
$\{v_1\} \subset \mathcal{M}^S$
(T3.2)
l_1
l_2
v_1
s_1
s_2
$\mathcal{G}_t - \mathcal{T}_j$
$\{l_1, l_2\} \subset \mathcal{I}$
$\{v_1\} \subset \mathcal{M}^S$
(T3.3)

Fig. 7. An example of the third case of assigning monitors in a tri-connected component.

The third case **T3** (line 30 to 36) is that $\mathcal{T}_j$ has two sep-vs s_1, s_2. Then OMA will first (**T3.1**) try to assign a vertex $v \in (V(\mathcal{T}) - \{s_1, s_2\})$ as a monitor, when v does not in any interesting link. Figure 7(T3.1) shows this case. OMA will assign v_1 as a monitor. If no such vertex (**T3.2**), OMA will try to assign a helper vertex v which is not in any interesting link as a monitor, when $h^{-1}(v)$ in $L(\mathcal{T}_j) - \{s_1 s_2\}$. Figure 7(T3.2) shows this case. OMA will assign the helper vertex v_1 as a monitor since it does not in any interesting link. Otherwise (**T3.3**), OMA will assign any vertex in $V(\mathcal{T}_j) - \{s_1, s_2\}$ as a monitor. Figure 7(T3.3) shows this case. OMA will assign v_1 as a monitor, but the identifiability of links which incident to v_1 depends on the assignment outside s_1, s_2.

C. Monitor Assignment for Cycles

Algorithm 3 assigns monitors for each cycle $\mathcal{T}_j$ in $\mathcal{G}_t$. It takes $\mathcal{T}_j, \mathcal{H}, \mathcal{I}, \mathcal{M}^S$ as input and updates the monitor assignment $\mathcal{M}^S$ when necessary.

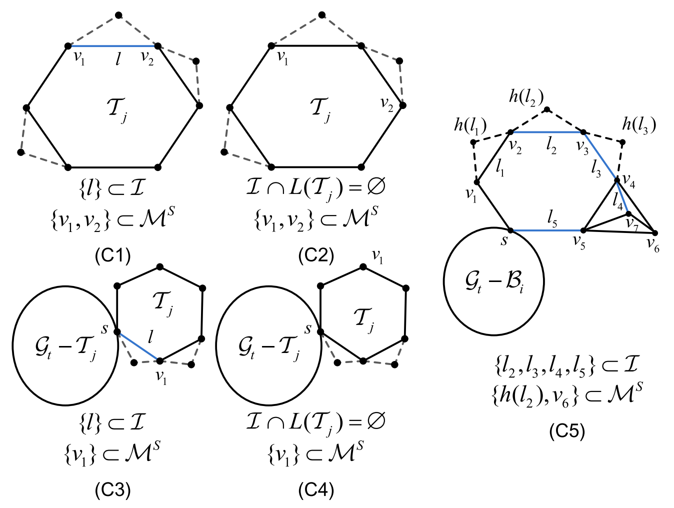

$\{l\} \subset \mathcal{I}$
$\{v_1, v_2\} \subset \mathcal{M}^S$
(C1)
$\mathcal{I} \cap L(\mathcal{T}_j) = \varnothing$
$\{v_1, v_2\} \subset \mathcal{M}^S$
(C2)
$\{l\} \subset \mathcal{I}$
$\{v_1\} \subset \mathcal{M}^S$
(C3)
$\mathcal{I} \cap L(\mathcal{T}_j) = \varnothing$
$\{v_1\} \subset \mathcal{M}^S$
(C4)
$\{l_2, l_3, l_4, l_5\} \subset \mathcal{I}$
$\{h(l_2), v_6\} \subset \mathcal{M}^S$
(C5)

Fig. 8. An example of assigning monitors in a cycle.

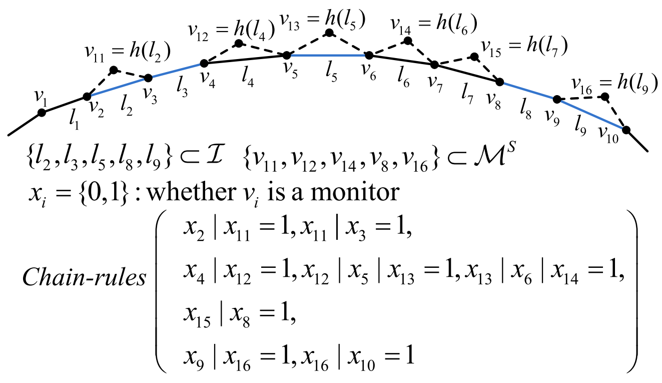

$\{l_2, l_3, l_5, l_8, l_9\} \subset \mathcal{I}$ $\{v_{11}, v_{12}, v_{14}, v_8, v_{16}\} \subset \mathcal{M}^S$
$x_i = \{0,1\}$: whether v_i is a monitor
Chain-rules
$x_2 \mid x_{11} = 1, x_{11} \mid x_3 = 1,$
$x_4 \mid x_{12} = 1, x_{12} \mid x_5 \mid x_{13} = 1, x_{13} \mid x_6 \mid x_{14} = 1,$
$x_{15} \mid x_8 = 1,$
$x_9 \mid x_{16} = 1, x_{16} \mid x_{10} = 1$

Fig. 9. An example of *Chain-rule*.

Algorithm 3 Minimum Monitor Assignment: Assign-Cycle

```
Input: Cycle $\mathcal{T}_j$, the helper vertex set $\mathcal{H}$, the interesting link set $\mathcal{I}$, the monitor assignment $\mathcal{M}^S$
Output: An updated monitor assignment $\mathcal{M}^S$
1: procedure ASSIGN-CYCLE($\mathcal{T}_j, \mathcal{H}, \mathcal{I}, \mathcal{M}^S$)
2:     if $\mathcal{T}_j$ has no sep-v, $L(\mathcal{T}_j) \cup \mathcal{I} = \{l\}$ then
3:         $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup V(l)$
4:     else if $\mathcal{T}_j$ has no sep-v, $L(\mathcal{T}_j) \cup \mathcal{I} = \emptyset$ then
5:         $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v_1, v_2\}, v_1, v_2 \in \mathcal{T}_j$
6:     else if $\mathcal{T}_j$ has one sep-v $s$, $L(\mathcal{T}_j) \cup \mathcal{I} = \{sv\}$ then
7:         $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v\}$
8:     else if $\mathcal{T}_j$ has one sep-v $s$, $L(\mathcal{T}_j) \cup \mathcal{I} = \emptyset$ then
9:         $\mathcal{M}^S \leftarrow \mathcal{M}^S \cup \{v\}, v \neq s, v \in \mathcal{T}_j$
10:    else
11:        for each $l \in \mathcal{I}(l = v_1 v_2)$ in the cycle do
12:            if $v_1$ is not a sep-v then
13:                generate Chain-rule at $v_1$'s side
14:            if $v_2$ is not a sep-v then
15:                generate Chain-rule at $v_2$'s side
16:        solve the Chain-rules with minimum monitors
```

First, there are four special cases needed to be considered (line 2 to 9). According to the number of sep-vs, the number of interesting links, and how the interesting link connects to the sep-v, these four special cases are described in the following. The first special case (**C1**) is that $\mathcal{T}_j$ has no sep-v and $\mathcal{T}_j$ has only one interesting link l. In this case, OMA assigns the two endpoints of l (i.e., $V(l)$) as monitors. Figure 8(C1) shows this case. OMA simply assigns v_1, v_2 as two monitors for identifying a single interesting link $l = v_1 v_2$. The second special case (**C2**) is that $\mathcal{T}_j$ has no sep-v and $\mathcal{T}_j$ has no interesting link. In this case, OMA randomly assigns two vertices in $\mathcal{T}_j$ as monitors. Figure 8(C2) shows this case. OMA simply assigns v_1, v_2 as two monitors. The third special case (**C3**) is that $\mathcal{T}_j$ has one sep-v s and $\mathcal{T}_j$ has only one interesting link sv. In this case, OMA assigns v as a monitor. Figure 8(C3) shows this case. OMA assigns v_1 as a monitor for identifying sv_1. The fourth special case (**C4**) is that $\mathcal{T}_j$ has one sep-v s and $\mathcal{T}_j$ has no interesting link. In this case, OMA randomly assigns a vertex (not s) in $\mathcal{T}_j$ as a monitor. Figure 8(C4) shows this case. OMA assigns v_1 as a monitor.

Then in general cases (**C5**, line 11 to 16), OMA assigns monitors for each interesting link $l = v_1 v_2$ in the cycle. For each vertex v_i of the two endpoints v_1, v_2, if v_i is not a sep-v, OMA generates the following rule at v_i's side for identifying l.

Chain-Rule: Let l' be the other link (not l) that incidents to v_i, one vertex in $\{h(l'), v_i, h(l)\}$ should be assigned as a monitor (or be a sep-v). Note that if one (or two) of the three vertices does not exist, the rule becomes that one vertex of the rest two (or one) vertices should be assigned as a monitor (or be a sep-v).

In Figure 8(C5), the *Chain-rule* for identifying link $l_2 = v_2 v_3$ at v_2's side is that one vertex in $\{h(l_1), v_2, h(l_2)\}$ should be assigned as a monitor. Similarly, the rule at v_3's side for identifying l_3 is that one vertex in $\{h(l_2), v_3, h(l_3)\}$ should be assigned as a monitor. Note that the above two rules include one common vertex $h(l_2)$. Connected by these common vertices, multiple *Chain-rules* form a chain, which is actually the reason that it is called as *Chain-rule*. Since v_4 is a sep-v, there is no *Chain-rule* generated at v_4's side for identifying l_3. After generating all the rules in the cycle, OMA solves the rules with minimum monitors (line 16).

In order to solve the *Chain-rules*, we have the following problem formulation.

Problem Formulation: Let $x_1, \ldots, x_k$ be a number of binary variables and each $x_i = \{0, 1\}$ represents whether a vertex is assigned as a monitor. Each *Chain-rule* is always in the form $x_a | x_b | x_c = 1$ or $x_a | x_b = 1$ and a pair of these constraints may have one overlapped variable. The problem is to find a solution of each x_i, when all constraints are satisfied and $\sum x_i$ reaches minimum.

Figure 9 shows an example of a number of *Chain-rule*. For example, a rule "$x_{12} | x_5 | x_{13} = 1$" is generated by OMA to identify link l_5. The eight rules shown in this example can be solved in four groups. Each group is one row of rules in the figure. The reason is that rules in different rows have no repeated variables, making them independent. For the first row, the optimal solution is $x_{11} = 1$ and $x_2 = x_3 = 0$. For a group with more than two dependent rules, we propose an efficient linear algorithm to solve it. The optimal solution of the four groups is to assign $v_{11}, v_{12}, v_{14}, v_8, v_{16}$ as monitors.

In the following, we focus on how to solve a group of *Chain-rules* efficiently. We use the second row of rules as an example.

- Step 1. Choose a variable x which represents a sep-v or a monitor as a start point, sort all variables according to their positions in the graph. A helper vertex v is in the middle of two endpoints of link $h^{-1}(v)$. In the second row of rules, we have the following sequence, $S_x = (x_4, x_{12}, x_5, x_{13}, x_6, x_{14})$.
- Step 2. Sort the rules according to the order in sequence S_x. In the example, the rules are reorganized as: $x_4|x_{12} = 1, x_{12}|x_5|x_{13} = 1, x_{13}|x_6|x_{14} = 1$ (the rules in the figure have been correctly sorted).
- Step 3. Let the last variable of the first rule be one, then remove the rules with that variable. In the example, the variable x_{12} is assigned to be 1 and the rules $x_4|x_{12} = 1, x_{12}|x_5|x_{13} = 1$ are removed.
- Step 4. Repeat step 3 for the rest rules till there is no rules left. In the example, x_{14} is assigned to be 1 in the second iteration.
- Step 5. The output is a binary vector $\mathcal{X}$, in which each entry is the value of the corresponding x_i in S_x. In the example, the output is $\mathcal{X} = (0, 1, 0, 0, 0, 1)$.

There is a special case when no sep-v or monitor in the cycle before solving the rules. In this case, the first step is different. OMA tries to assign a random vertex (which is an endpoint of an interesting link) or one of its two adjacent helper vertices (when exist) as a monitor. Then OMA goes through the above algorithm and obtains at most three solutions. The solution with the minimum monitors is the final solution. The proof of the optimality of this algorithm is included in the technical report of OMA [33].

D. Boundary Cases for Bi-Connected Components

In the algorithm sketch of the monitor assignment (Algorithm 1), OMA assigns monitors in each SPQR component (line 5 to 8) in each bi-connected component (line 3). After that, there are still some boundary cases of each bi-connected component need to be handled with. An example of these boundary cases is that when $\mathcal{G}_t$ is only a single interesting link. In that case, it is neither a tri-connected component nor a cycle. Therefore, Algorithm 2 (handling tri-connected components) and Algorithm 3 (handling cycles) will not assign any monitors for this $\mathcal{G}_t$. The optimal monitor assignment for this boundary case is simple, which is to assign the two vertices as monitors. Due to the page limit, the detailed description of handling these boundary cases is included in the technical report of OMA [33].

E. Time Complexity of OMA

Theorem 3: The time complexity of OMA is linear in terms of the number of vertices and links, i.e., O($|V(\mathcal{G})| + |L(\mathcal{G})|$).

The proof of Theorem 3 is included in the technical report of OMA [33].

V. Proof of Optimality

In this section, we theoretically prove the optimality of OMA. OMA assigns a number of monitors in $V(\mathcal{G}_t) \cup \mathcal{H}$ after graph trimming. We use $\mathcal{M}^S(\mathcal{G}_t \cup \mathcal{H})$ to denote this assignment. Formally, the following theorem describes the optimality of OMA.

Theorem 4: The monitor assignment $\mathcal{M}^S(\mathcal{G}_t \cup \mathcal{H})$ is one optimal monitor assignment of the original graph $\mathcal{G}$.

In [8], it has been proven that an optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ is also an optimal monitor assignment $\mathcal{M}^*(\mathcal{G})$ ([8, Th. VI.2]). Therefore, we focus on proving the following theorem.

Theorem 5: The monitor assignment $\mathcal{M}^S(\mathcal{G}_t \cup \mathcal{H})$ is one optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$.

In order to prove Theorem 5, we first prove the following two theorems.

Theorem 6: For any connected graph $\mathcal{G}$, the following equation always holds.

$$|\mathcal{M}^S(\mathcal{G}_t \cup \mathcal{H})| \leq |\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})|. \tag{1}$$

Theorem 7: The monitor assignment $\mathcal{M}^S(\mathcal{G}_t \cup \mathcal{H})$ is able to identify all interesting links in $\mathcal{G}$.

We will prove the above two theorems later in this section. Given these two theorems, we can prove Theorem 5 formally as follows.

Proof of Theorem 5: According to Theorem 7, the monitor assignment $\mathcal{M}^S(\mathcal{G}_t \cup \mathcal{H})$ is able to identify all interesting links in $\mathcal{G}$. Therefore, $|\mathcal{M}^S(\mathcal{G}_t \cup \mathcal{H})| \geq |\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})|$. Combining this with Equation 1, the following equation holds.

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

Since the monitor assignment $\mathcal{M}^S(\mathcal{G}_t \cup \mathcal{H})$ has the same number of monitors compared with the optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$ and it is able to identify all interesting links in $\mathcal{G}$ (Theorem 7), it is an optimal monitor assignment $\mathcal{M}^*(\mathcal{G}_t \cup \mathcal{H})$. □

Therefore, the key to prove the optimality of OMA is to prove Theorem 6 and Theorem 7. In the following, we first give some results from existing work and then prove these two theorems separately.

A. Existing Results

Before proving the optimality of OMA, we first give some important results from existing work [3], [25].

Theorem 8: If $\mathcal{G}$ is a tri-connected graph, all of its link metrics are identifiable by assigning any three monitors.

Corollary 9: If $\mathcal{T}$ is a tri-connected component of a graph $\mathcal{G}$, 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) one or more vertices v_i is not in $\mathcal{T}$, but there are distinct paths (without any repeated vertex) from v_i to $\mathcal{T}$.

*Definition 10: 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 11: For a tri-connected component $\mathcal{T}$ with two monitors assigned in it (or two sep-vs which connect to

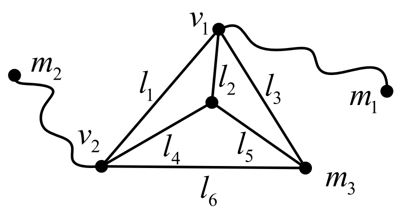


Fig. 10. A typical tri-connected graph to illustrate several results from previous works. l_2 is an interior link w.r.t. vertices v_2, m_3. And l_2 can be identified by two monitors $\{v_2, m_3\}$ or $\{m_2, m_3\}$.

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 10 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 8 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 9 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 11 says if we assign two monitors v_2, m_3, the interior links (i.e. l_2) can be identified and the exterior links (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.

B. Proof of Theorem 6

Theorem 6 says $|\mathcal{M}^S|$ should be less than or equal to $|\mathcal{M}^*(V(\mathcal{G}_t \cup \mathcal{H}))|$, we will prove it by proving that in each graph component (i.e., case T1 to T3, case C1 to C5 and case B1 to B3), the number of monitors assigned by OMA *cannot* be reduced, otherwise at least one interesting link cannot be identified. The analysis of the boundary cases are included in the technical report of OMA [33].

In case **T1**, the graph $\mathcal{G}_t$ is a tri-connected graph. Since one monitor cannot identify any link, at least two monitors are required to be assigned (i.e., two monitors are minimum for identifying any interesting links). In case **T1.1**, **T1.2** and **T1.3**, there are only two monitors assigned. Therefore, the number of assigned monitors cannot be reduced. In case **T1.4**, OMA assigns three monitors. We then show that two monitors cannot identify all interesting links. In Algorithm 2, we have defined two vertex set V_1, V_2. In V_1, each vertex is in $\mathcal{T}_j$ and is not in any interesting link in $\mathcal{T}_j$. In V_2, each vertex is a helper vertex of a link in $\mathcal{T}_j$ and does not in any interesting link. In T1.4, there are two possibilities, $|V_1| + |V_2| < 2$ or $|V_1| = |V_2| = 1$. If $|V_1| + |V_2| < 2$, we cannot find two monitors that are not in any interesting links. In [34], it has been proven that when there are only two monitors, all links that incident to only one of the two monitors cannot be identified. Therefore, two monitors cannot identify all interesting links when $|V_1| + |V_2| < 2$. If $|V_1| = |V_2| = 1$ ($V_1 = \{v_1\}, V_2 = \{v_2\}$), v_1 must be one endpoint of $V(h^{-1}(v_2))$ (line 12 in Algorithm 2). Since $V_1 = \{v_1\}$, the other endpoint of $V(h^{-1}(v_2))$ must be in at least one interesting link l. According to Theorem 11, l cannot be identified. Therefore, assigning two monitors in case T1.4 cannot identify all interesting links, which means the number of monitors in $|\mathcal{M}^S|$ cannot be reduced in case T1.4.

In case **T2.1.1**, **T2.1.2** and **T2.2**, OMA only assigns one monitor in the tri-connected component $\mathcal{T}_j$. Since there is only one sep-v in $\mathcal{T}_j$, no measurement path can be conducted in $\mathcal{T}_j$ without that monitor, causing all links in $\mathcal{T}_j$ be not identifiable. In case **T2.1.3**, all vertices in $\mathcal{T}_j$ are in at least one interesting link and all links associated to the helper vertices incident to the 1-cut-v s. If we assign a helper vertex v_1 as monitor, the links that incident to $V(h^{-1}(v_1))$ are not identifiable. If we assign a vertex v_2 in $\mathcal{T}_j$ as a monitor, all links that incident to v_2 (not $v_2 s$) are unidentifiable. Since there is at least one interesting link that incidents to $V(h^{-1}(v_1))$ or v_2 (line 19 and 21 in Algorithm 2), assigning v_1 or v_2 cannot identify this interesting link. Therefore, one monitor cannot identify all interesting links. In case **T2.3**, there are more than one interesting links which incident to the 1-cut-v s, so there will still be at least one exterior interesting link after assigning one monitor in $\mathcal{T}_j - s$. According to Theorem 11, this interesting link cannot be identified by assigning one monitor in $\mathcal{T}_j - s$. Therefore, the number of monitors in $|\mathcal{M}^S|$ cannot be reduced in case T2.3.

In case **T3**, OMA only assigns one monitor in the tri-connected component $\mathcal{T}_j$. If no monitor is assigned in $\mathcal{T}_j$, all exterior links will not be identifiable. According to the graph trimming algorithm [8], $\mathcal{T}_j$ includes at least one interesting exterior link. Therefore, the monitor assigned in $\mathcal{T}_j$ is necessary.

In case **C1** and **C2**, the trimmed graph $\mathcal{G}_t$ is a cycle. Since one monitor cannot identify any link, at least two monitors are required to be assigned (i.e., two monitors are minimum for identifying any interesting links). In case C1 and C2, only two monitors are assigned. Therefore, the number of monitors in $|\mathcal{M}^S|$ cannot be reduced in case C1 and C2. In case **C3** and **C4**, only one monitor is assigned. Due to the same reason of case T2.1.1, the number of monitors in $|\mathcal{M}^S|$ cannot be reduced in case C4.

In case **C5**, OMA solves the optimization problem with multiple *Chain-rules* to assign monitors. If a rule does not hold, we explain why at least one interesting link cannot be identified as follows. Without loss of generality, the *Chain-rule* at v_2's side to identify l_2 in Figure 8(C5) is a typical example. Its *Chain-rule* is to assign one monitor in $\{h(l_1), v_2, h(l_2)\}$. Its opposite is not to assign any monitor in $\{h(l_1), v_2, h(l_2)\}$. Then we prove that the rule is necessary by contradiction. Assume no vertex in $\{h(l_1), v_2, h(l_2)\}$ is assigned as a monitor and l_2 is still identifiable by a certain monitor assignment $\mathcal{M}'$. According to [8, Th. VI.6], assigning v_1, v_3 as monitors instead of $\mathcal{M}'$ can also identify l_2. However, l_2 cannot be identified by v_1, v_3 since l_2 incidents to v_3 [3]. This contradicts the assumption. Similarly, all rules are necessary for identifying interesting links in a cycle. Therefore, the number of monitors in $|\mathcal{M}^S|$ cannot be reduced in case C5.

According to the above analysis, we can see that the number of monitors assigned in $|\mathcal{M}^S|$ cannot be reduced in any case. Therefore, Theorem 6 always holds.

C. Proof of Theorem 7

Before proving Theorem 7, we first introduce the following lemmas.

Lemma 12: Let a link $l = v_1v_2$ in a bi-connected graph (or component) $\mathcal{B}$, the link must be in at least one SPQR component after the SPQR partitioning of $\mathcal{B}$.

Lemma 13: Given a bi-connected graph $\mathcal{B}$ (not a single SPQR component), there must be at least two SPQR components which have only two 2-cut-vs, after conducting SPQR partitioning of $\mathcal{B}$.

Lemma 14: For a bi-connected component $\mathcal{B}$, there are two cases after using OMA to assign monitors for $\mathcal{B}$. 1) The number of 1-cut-v and monitors in $\mathcal{B}$ is two, and all interesting links in $\mathcal{B}$ (including those been trimmed) are identifiable. 2) The number of 1-cut-vs and monitors in $\mathcal{B}$ is larger than two.

The proofs of the above lemmas are included in the technical report of OMA [33]. Based on the above lemmas, we are ready to prove Theorem 7.

Proof of Theorem 7: For each bi-connected component $\mathcal{B}$ in $\mathcal{G}_t$, there are two cases in terms of the number of monitors assigned and 1-cut-vs (Theorem 14). 1) The number of monitors assigned and 1-cut-vs is two and all interesting links in $\mathcal{B}$ (including those been trimmed) are identifiable. 2) The number of monitors assigned and 1-cut-vs is larger than two. Since all interesting links in case 1) have been proven to be identifiable, we only need to consider case 2). In particular, we need to prove that for each bi-connected component in which *the number of monitors and 1-cut-vs is at least three*, all its interesting links (including those been trimmed) are identifiable.

Consider an interesting link l which is in $V(\mathcal{G} - \mathcal{G}_t)$, it is identifiable in both of the following two cases. If the two endpoints of l are monitors, l is obviously identifiable. If at most one of the endpoints of l is a monitor, there must be at least two monitors/1-cut-vs which are not endpoints of l. The reason is that we only need to consider the bi-connected component with at least three monitors/1-cut-vs, as described in the previous paragraph. According to Theorem 11, l can be identified in this case. Therefore, all interesting links in $V(\mathcal{G} - \mathcal{G}_t)$ can be identified.

Then we consider the identifiability of interesting links in each SPQR component of $\mathcal{G}_t$. Since an SPQR component can be a tri-connected component or a cycle, we analyze the identifiability separately.

When the SPQR component is a tri-connected component, there are four cases, i.e., T1 (zero sep-v), T2 (one sep-v), T3 (two sep-vs) in Algorithm 2, and T4 in which there are at least three sep-vs of the tri-connected component. In case **T1**, $\mathcal{G}_t$ is a tri-connected component. Since it has no 1-cut-v, there must be at least three monitors assigned in $\mathcal{G}_t \cup \mathcal{H}$. When these monitors are in $\mathcal{G}_t$, all links in $\mathcal{G}_t$ are identifiable (Theorem 8). When one or more helper vertices are assigned as monitors, they have distinct paths to $\mathcal{G}_t$. According to Corollary 9, all links in $\mathcal{G}_t$ are identifiable. In case **T2** (or **T3**), the tri-connected component $\mathcal{T}$ has one (or two) 1-cut-v(s). Then there must be at least two (or one) monitors assigned in it. Similar to case T1, all links in $\mathcal{T}$ are identifiable according to Corollary 9. In case **T4**, the tri-connected component has at least three sep-vs. According to Corollary 9, all links in it are identifiable. Therefore, all interesting links in $\mathcal{G}_t$ are identifiable in case the SPQR component is a tri-connected component.

When the SPQR component is a cycle, there are five cases, i.e., C1 to C5 in Algorithm 3. Since there is no interesting link in the cycle in case **C2** and **C4**, we only need to consider case C1, C3 and C5. In case **C1**, the two endpoints of the only interesting link l are assigned as monitors. Therefore, l is obviously identifiable. In case **C3**, the cycle $\mathcal{C}$ has one 1-cut-v s and one interesting link sv. In this case, OMA will assign v as a monitor (line 7 in Algorithm 3). Then there are already one 1-cut-v (i.e., s) and one monitor (i.e., v), there must be at least one additional monitor m in the cycle (since *the number of monitors and 1-cut-vs is at least three*). Then a monitor m' in $\mathcal{G}_t - \mathcal{C}$, monitor v, monitor m and path $s \sim m'$, path/link sv, path $s \sim m$ form a "Y"-like structure. Three measurement paths between each of the two monitors can easily identify these three paths. Therefore, link sv is identifiable in case C3. In case **C5**, OMA solves the *Chain-rules* to assign monitors. Consider an interesting link s_1s_2 in case C5, OMA will generate a *Chain-rule* for each of the two endpoints, if the endpoint is not a sep-v (line 12, 14 in Algorithm 3). Without loss of generality, we take s_1 as an example. Let l_1 is another link that incidents to s_1 in the same cycle. Then the *Chain-rule* of s_1 is that there should be one monitor in the following three vertices (if exist): s_1, the helper vertex of l_1 and the helper vertex of s_1s_2. Then according the vertices assigned as monitors for s_1s_2, there are five different cases. First, s_1, s_2 are assigned as monitors. s_1s_2 is obviously identifiable in this case. Second, the helper vertex of s_1s_2 is assigned as a monitor and there are two monitors which have distinct paths to s_1s_2. According to Corollary 9, s_1s_2 is identifiable. Third, the helper vertex of l_1 is assigned as a monitor and s_2 is assigned as a monitor. Similar to case C3, s_1s_2 can be identified by a "Y"-like structure. Fourth, two helper vertices of the links adjacent to s_1s_2 are assigned as monitors. In this case, s_1s_2 can be identified by using the "Y"-like structure twice. Fifth, one or two endpoints of s_1s_2 are sep-vs. For example, s_1 is a sep-v. In this case, there must be one monitor assigned in the adjacent SPQR components with s_1. Then similar to the third and fourth cases, link s_1s_2 is already identifiable without assigning monitors by solving the *Chain-rules*. Therefore, each interesting link in case C5 is identifiable.

Since the interesting links in tri-connected components and cycle are all identifiable, all interesting links of SPQR components are identifiable, for each bi-connected component in which *the number of monitors and 1-cut-vs is at least three*. Therefore, all interesting links are identifiable by the monitor assignment obtained by OMA. □

VI. Evaluation

We evaluate OMA in both real network topologies and synthetic topologies. In this section, we first give the evaluation methodology. Then we give the results in real

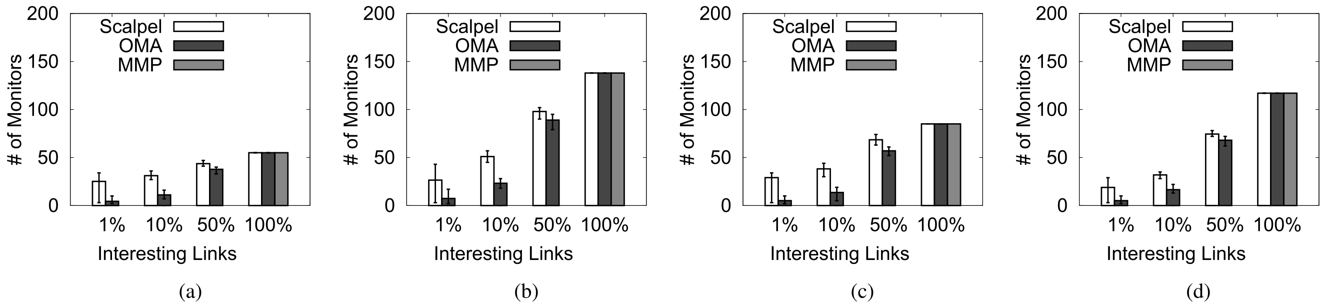


Fig. 11. Monitor assignment results in four real network topologies from the Rocketfuel project. When all links are interesting links (i.e., 100% in the figure), the monitor assignment results of OMA, MMP and Scalpel are all the same. Since MMP always views all links as interesting links, we only show the 100% case which represents the performance of MMP in all cases. (a) AS1755. (b) AS3257. (c) AS3967. (d) AS6461.

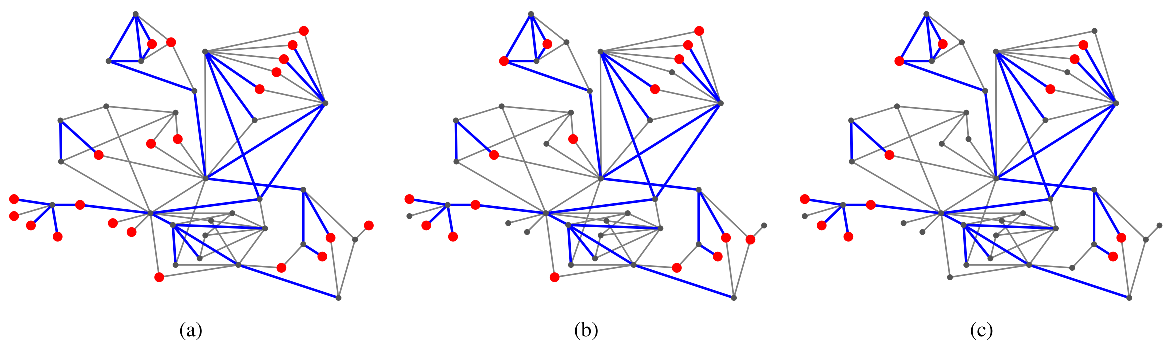


Fig. 12. The topology and the monitors assigned in part of a network from the Rocketfuel project. Blue links are interesting links. Red vertices are monitors assigned by the three approaches. (a) MMP. (b) Scalpel. (c) OMA.

network topologies. Finally, we study the performance of OMA in various network settings.

A. Methodology

The evaluation metric is the number of monitors assigned. Two baseline approaches are implemented and compared with OMA. First, we implement MMP [3], which assigns a minimum number of monitors to identify *all* links in a network. Therefore, MMP is an extreme case of the preferential link tomography problem in which all links are interesting links. Second, we implement Scalpel, a non-optimal monitor assignment approach which also focuses on preferential link tomography.

Two kinds of topologies are used to evaluate the monitor assignment performance of OMA, MMP, and Scalpel, real network topologies and synthetic network topologies. We use three datasets of real internet topologies, topologies from the Rocketfuel [35] project, the IDTK [36] from CAIDA, and the AS-rank project [37] from CAIDA. In order to compare the performance of OMA with MMP using the same topologies, we include the Rocketfuel topologies in the evaluation. Since the Rocketfuel topologies are obtained about a decode ago and modern network topologies have evolved, we also use more recent router-level topologies from the IDTK project. In addition to router-level topologies, we also use AS-level topologies to evaluate the performance of OMA. We use four AS-level topologies from the AS-rank project. For the topologies from the IDTK project and the AS-rank project, we use *dK*-graph [38] to scale them down.

Further, we use synthetic topologies to evaluate the monitor assignment performance of the OMA and two baseline approaches. Specifically, we study the impact of the topology type and the impact of the network scale. In order obtain these various network topologies, we use a network topology generator, *dK*-graph [38], to generate them. Some links are randomly designated as interesting links in each experiment. We repeat each experiment 10 times and report the average value as well as the maximum/minimum values.

B. Real Topologies

We use twelve real topologies to evaluate the monitor assignment performance of OMA, MMP, and Scalpel. In these twelve topologies, four of them are from the Rocketfuel projects, four of them are from the IDTK project (down scaled), and four of them are from the AS-rank project (down-scaled). Table I gives the number of nodes and links of these topologies.

Figure 11 shows the monitor assignment results in the four Autonomous Systems (AS) from the Rocketfuel [35] project. The x-axis is the ratio of interesting links in all links, from 1% to 100%. The y-axis is the number of monitors assigned. When there are more interesting links, more monitors are assigned by OMA and Scalpel. Since MMP always views all links as interesting links (i.e., the 100% case), the number of monitors assigned does not depend on the ratio of interesting links. Therefore, we only show the 100% case for MMP. Compared with Scalpel, OMA reduces the number of monitors assigned by 37.7% to 83.3% when 10% of the links are interesting links.

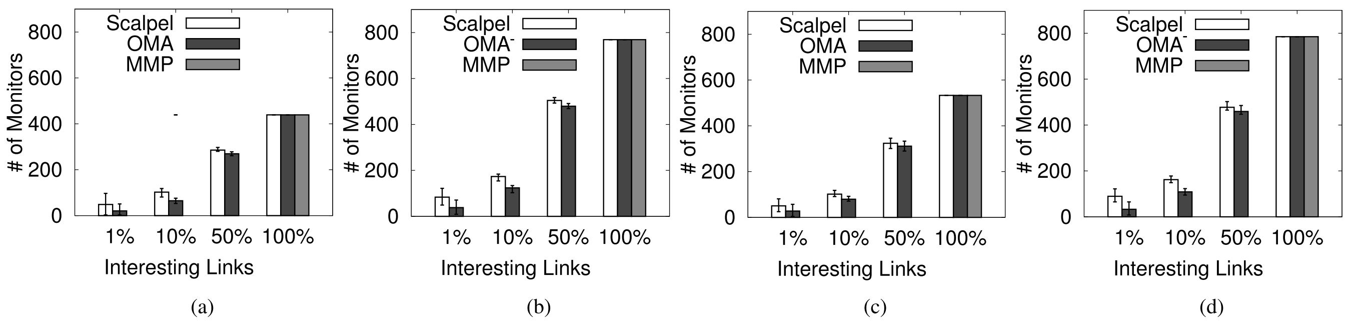


Fig. 13. Monitor assignment results in four router-level network topologies from the CAIDA IDTK project. (a) AS1680. (b) AS3758. (c) AS6057. (d) AS8220.

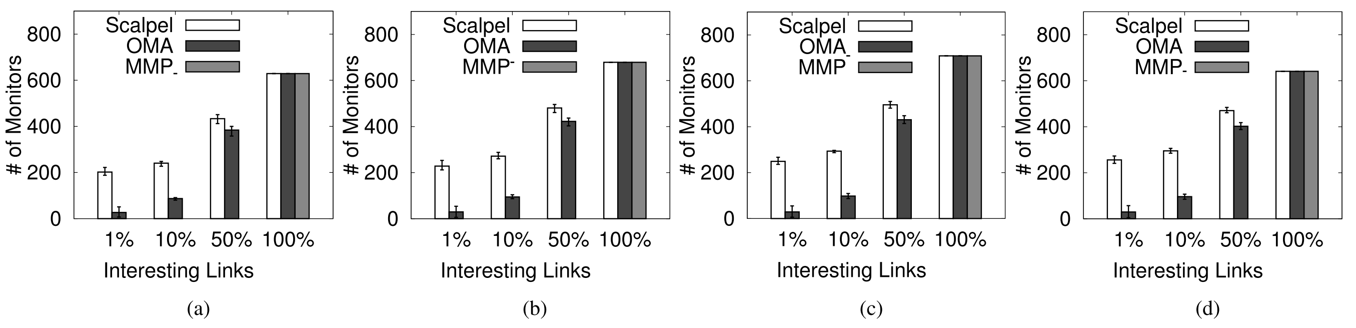


Fig. 14. Monitor assignment results in four AS-level network topologies from the CAIDA AS-rank project. (a) 20151001. (b) 20151101. (c) 20151201. (d) 20160101.

TABLE I
NUMBER OF NODES AND LINKS OF THE TWELVE TOPOLOGIES

Topology	Project	# of nodes	# of links
AS1755	Rocketfuel	172	381
AS3257	Rocketfuel	240	404
AS3967	Rocketfuel	201	434
AS6461	Rocketfuel	182	294
AS1680	CAIDA IDTK	717	1792
AS3758	CAIDA IDTK	1169	2205
AS6057	CAIDA IDTK	782	1289
AS8220	CAIDA IDTK	1168	2233
20151001	CAIDA AS-rank	869	1483
20151101	CAIDA AS-rank	924	1623
20151201	CAIDA AS-rank	987	1828
20160101	CAIDA AS-rank	944	1791

Figire 12 shows the topology and the monitor assignment in part of a network from the Rocketfuel project. 40% of the links are interesting links. The monitors assigned by the three approaches are shown in the figure. MMP assigns 22 monitors, Scalpel assigns 17 monitors and OMA assigns 12 monitors.

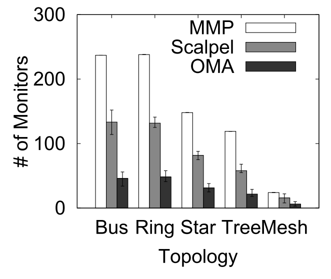
We also use more recent topologies from two projects of CAIDA. Figure 13 shows the evaluation results in the four topologies from the CAIDA IDTK [36] project. Compared with the topologies from the Rocketfuel projects, these networks include more nodes and links. OMA assigns the minimum number of monitors, reducing the monitors by 21.9% to 36.7% compared with Scalpel, when 10% of the links are interesting links.

The above topologies are all router-level topologies. We further evaluate the performance of OMA using AS-level topologies. Figure 14 shows the evaluation results in the four AS topologies from the CAIDA AS-rank [37] project. Compared with the results in Figure 13, we can see that Scalpel assigns much more monitors than OMA when 10% of the links are interesting links. Take router-level topology AS1680 and AS-level topology 20151001 as an example, Scalpel assigns 102.4 and 240.4 monitors on average, respectively. When we use OMA to assign monitors, only 64.8 and 86.9 monitors are required to identify all the interesting links on average. In all cases, OMA assigns the minimum number of monitors, reducing the monitors by 63.9% to 67.4% compared with Scalpel, when 10% of the links are interesting links.

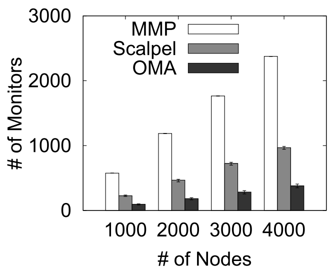
C. Synthetic Topologies

We use synthetic topologies generated by the dK-graph [38] tool to study the impact of the topology type and the network scale.

Impact of Topology Type: Figure 15(a) shows the monitor assignment results in the five networks with different topology types. These five networks include the same number of nodes (300 nodes) and different number of links (from 448 links to 932 links). Among those links, 10% of them are interesting links. We can see that the bus topology and the ring topology need more monitors to identify the interesting links and the mesh topology needs the least monitors. The reason is that there are more links in the mesh topology than in the other topologies. Since more links can enable more measurement paths between pairs of monitors, the mesh topology requires less monitors than the other four topologies. Note that real ISP network topologies could be much more complicated



(a)



(b)

Fig. 15. Impact of topology type and network scale to the monitor assignment performance. (a) Impact of topology. (b) Impact of network scale.

(e.g., hybrid design, multi-layered structures). For example, the topology shown in Figure 12 is a part of a real network topology from the Rocketfuel project. This topology is none of the five basic network topology type, but a hybrid topology. Therefore, we only use these synthetic network topologies to evaluate the performance of OMA in extreme cases (i.e., pure topology types).

Impact of Network Scale: Figure 15(b) shows the results in four networks with different number of nodes. These four topologies are re-scaled by the dK-graph [38] tool from the same real network topology, they have the same network type. As shown in the figure, all the three approaches require more monitors in larger networks. When 10% of the links are interesting links, OMA reduces the number of monitors by 82.3% to 86.2% compared with MMP and 55.5% to 63.5% compared with Scalpel.

VII. Conclusion

In this paper, we propose OMA, an optimal monitor assignment algorithm for preferential link tomography. Given a graph representing a network topology and a set of interesting links, OMA first partitions the graph into different components and uses a graph trimming algorithm to trim the graph. Then OMA carefully assigns vertices as monitors for each of the graph components to achieve both the identifiability of interesting links and the minimum number of monitors. Since the identifiability of interesting links in different graph components are not isolated, OMA uses a *Chain-rule* formulation to model this dependency and solves it by a novel linear algorithm. We theoretically prove the optimality of OMA and evaluate its performance in both real network topologies and synthetic topologies. Compared with two baseline approaches, OMA significantly reduces the number of monitors assigned in various network settings.

References

[1] E. Lawrence, G. Michailidis, V. N. Nair, and B. Xi, "Network tomography: A review and recent developments," in *Frontiers in Statistics*. London, U.K.: Imperial College Press, 2006, pp. 345–364.

[2] R. Kumar and J. Kaur, "Practical beacon placement for link monitoring using network tomography," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 12, pp. 2196–2209, Dec. 2006.

[3] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Inferring link metrics from end-to-end path measurements: Identifiability and monitor placement," *IEEE/ACM Trans. Netw.*, vol. 22, no. 4, pp. 1351–1368, Aug. 2014.

[4] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," *J. Amer. Statist. Assoc.*, vol. 91, no. 433, pp. 365–377, 1996.

[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, Jun. 2012.

[6] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM J. Comput.*, vol. 1, no. 2, pp. 146–160, 1972.

[7] G. Di Battista and R. Tamassia, "On-line graph algorithms with SPQR-trees," in *Automata, Languages and Programming*. Berlin, Germany: Springer-Verlag, 1990, pp. 598–611.

[8] Y. Gao *et al.*, "Scalpel: Scalable preferential link tomography based on graph trimming," *IEEE/ACM Trans. Netw.*, vol. 24, no. 3, pp. 1392–1403, Jun. 2015.

[9] A. B. Downey, "Using pathchar to estimate Internet link characteristics," in *Proc. ACM SIGCOMM*, 1999, pp. 241–250.

[10] G. Jin, G. Yang, B. R. Crowley, and D. A. Agarwal, "Network characterization service (NCS)," in *Proc. IEEE HPDC*, Aug. 2001, pp. 289–299.

[11] 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.

[12] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu, "Network tomography: Recent developments," *Statist. Sci.*, vol. 19, no. 3, pp. 499–517, 2004.

[13] T. Bu, N. Duffield, F. L. Presti, and D. Towsley, "Network tomography on general topologies," in *Proc. ACM SIGMETRICS*, 2002, pp. 21–30.

[14] A. Chen, J. Cao, and T. Bu, "Network tomography: Identifiability and Fourier domain estimation," *IEEE Trans. Signal Process.*, vol. 58, no. 12, pp. 6029–6039, Dec. 2010.

[15] T. He *et al.*, "Fisher information-based experiment design for network tomography," in *Proc. ACM SIGMETRICS*, 2015, pp. 389–402.

[16] R. Caceres, N. G. Duffield, J. Horowitz, and D. F. Towsley, "Multicast-based inference of network-internal loss characteristics," *IEEE Trans. Inf. Theory*, vol. 45, no. 7, pp. 2462–2480, Nov. 1999.

[17] 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, Oct. 2006.

[18] 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, Feb. 2009.

[19] 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.

[20] O. Gurewitz and M. Sidi, "Estimating one-way delays from cyclic-path delay measurements," in *Proc. IEEE INFOCOM*, Apr. 2001, vol. 2, pp. 1038–1044.

[21] S. S. Ahuja, S. Ramasubramanian, and M. Krunz, "SRLG failure localization in all-optical networks using monitoring cycles and paths," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 700–708.

[22] 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.

[23] 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.

[24] S. Tati, S. Silvestri, T. He, and T. La Porta, "Robust network tomography in the presence of failures," in *Proc. IEEE ICDCS*, Jun./Jul. 2014, pp. 481–492.

[25] 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.

[26] 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.

[27] *RCF0791 (Internet Protocol)*. Sep. 1981. [Online]. Available: https://tools.ietf.org/html/rfc791

[28] X. Yang and D. Wetherall, "Source selectable path diversity via routing deflections," in *Proc. ACM SIGCOMM*, 2006, pp. 159–170.

[29] M. Motiwala, M. Elmore, N. Feamster, and S. Vempala, "Path splicing," in *Proc. ACM SIGCOMM*, 2008, pp. 27–38.

[30] H. Abu-Libdeh, P. Costa, A. Rowstron, G. O'Shea, and A. Donnelly, "Symbiotic routing in future data centers," in *Proc. ACM SIGCOMM*, 2010, pp. 51–62.

[31] M. Soliman, B. Nandy, I. Lambadaris, and P. Ashwood-Smith, "Exploring source routed forwarding in SDN-based WANs," in *Proc. ICC*, 2014, pp. 3070–3075.

[32] S. Hu *et al.*, “Explicit path control in commodity data centers: Design and applications,” in *Proc. USENIX NSDI*, 2015, pp. 15–28.

[33] *Optimal Monitor Assignment for Preferential Link Tomography in Communication Networks [Tech-Report]*, accessed on Jun. 23, 2016. [Online]. Available: https://www.dropbox.com/s/35gdib3nklg6l19/link-optimal-tech.pdf

[34] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, “Link identifiability in communication networks with two monitors,” in *Proc. IEEE GLOBECOM*, Dec. 2013, pp. 1513–1518.

[35] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, “Measuring ISP topologies with Rocketfuel,” *IEEE/ACM Trans. Netw.*, vol. 12, no. 1, pp. 2–16, Feb. 2004.

[36] *The CAIDA AS Relationships Dataset, 20151001-20160101*, accessed on Jun. 23, 2016. [Online]. Available: http://www.caida.org/data/active/as-relationships/

[37] *The CAIDA UCSD Internet Topology Data Kit—2013-07*, accessed on Jun. 23, 2016. [Online]. Available: http://www.caida.org/data/active/internet-topology-data-kit

[38] P. Mahadevan, C. Hubble, D. Krioukov, B. Huffaker, and A. Vahdat, “Orbis: Rescaling degree correlations to generate annotated Internet topologies,” in *Proc. ACM SIGCOMM*, 2007, pp. 325–336.

Wei Dong (S’08–M’12) received the B.S. and Ph.D. degrees in computer science from Zhejiang University, Hangzhou, China, in 2005 and 2011, respectively. He is currently an Associate Professor with the College of Computer Science, Zhejiang University. His research interests include networked embedded systems, network measurement, and wireless sensor networks.

Yi Gao (M’15) received the B.S. and Ph.D. degrees from Zhejiang University, China, in 2009 and 2014, respectively. From 2011 to 2012, he visited McGill University as a Research Training Student. He is currently a Research Assistant Professor with Zhejiang University. His research interests include network measurement, wireless sensor networks, and mobile computing. He is a member of the IEEE and the ACM.

Wenbin Wu received the B.S. degree from the Zhejiang University of Technology in 2012, and the M.S. degree from Zhejiang University in 2015. His research interests include fine-grained measurement in wireless sensor networks and network tomography techniques in modern communication networks.

Jiajun Bu (M’06) received the B.S. and Ph.D. degrees in computer science from Zhejiang University, China, in 1995 and 2000, respectively. He is currently a Professor with the College of Computer Science and the Deputy Director of the Institute of Computer Software with Zhejiang University. His research interests include embedded system, mobile multimedia, and data mining. He is a member of the IEEE and the ACM.

Chun Chen (M’08) received the B.Math. degree from Xiamen University, China, in 1981, and the M.S. and Ph.D. degrees in computer science from Zhejiang University, China, in 1984 and 1990, respectively. He is currently a Professor with the College of Computer Science, and also an Academician with the Chinese Academy of Engineering. His research interests include embedded system, image processing, computer vision, and CAD/CAM.

Xiang-Yang Li (SM’08–F’15) received the bachelor’s degrees from the Department of Computer Science and the Department of Business Management, Tsinghua University, China, in 1995, and the M.S. and Ph.D. degrees from the Department of Computer Science, University of Illinois at Urbana–Champaign, in 2000 and 2001, respectively. He was a Professor with the Illinois Institute of Technology. He is currently a Professor and an Executive Dean of the School of Computer Science and Technology with the University of Science and Technology of China. He has authored a monograph entitled *Wireless Ad Hoc and Sensor Networks: Theory and Applications* and co-edited several books, including *Encyclopedia of Algorithms*. His research interests include wireless networking, mobile computing, security and privacy, cyber physical systems, and algorithms. He is a recipient of the China NSF Outstanding Overseas Young Researcher (B). He was an IEEE Fellow (2015) and an ACM Distinguished Scientist (2015). He holds the EMC-Endowed Visiting Chair Professorship at Tsinghua University from 2014 to 2016. He has received the six best paper awards, and one best demo award. He is an Editor of several journals and has served many international conferences in various capacities.