Preferential Link Tomography in Dynamic Networks

Huikang Li, Yi Gao*, Wei Dong, and Chun Chen
College of Computer Science, Zhejiang University
Email: {lihk, gaoyi, dongw, chenc}@zju.edu.cn

***Abstract*—Inferring fine-grained link metrics by using aggregated path measurements, known as *network tomography*, is essential for various network operations, such as network monitoring, load balancing, and failure diagnosis. Given a set of *interesting links* and the changing topologies of a dynamic network, 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. We propose MAPLink, an algorithm that assigns a number of nodes as monitors to solve this tomography problem. As the first algorithm to solve the preferential link tomography problem in dynamic networks, MAPLink guarantees that the assigned monitors can calculate the link metrics of all interesting links for all topologies of the dynamic network. We formally prove the above property of MAPLink based on graph theory. We implement MAPLink and evaluate its performance using two real-world dynamic networks, including a vehicular network and a sensor network, both with changing topologies due to node mobility or wireless dynamics. Results show that MAPLink achieves significant better performance compared with three baseline methods in both of the two dynamic networks.**

I. Introduction

Accurate and timely knowledge of internal network states (e.g., delays on individual links) is essential for various network operations such as route selection, resource allocation, and fault diagnosis. However, directly measuring the performance of such link metrics is not always feasible due to the measurement overhead, the lack of monitoring capability at internal network elements (e.g., switches, routers), and etc. *Network tomography* [1], instead, focuses on using *end-to-end* measurements to infer *hop-by-hop* link metrics, providing an attractive approach to calculate the internal link metrics.

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. Consider additive link metrics, e.g., delays or packet loss ratios after applying the log() function, the end-to-end measurement is the sum of individual link metrics along the measurement path. Then by solving a system of equations, the unknown link metrics can be calculated. Clearly, if all nodes are assigned as monitors, the metrics of all individual links can be easily calculated (i.e., all links are *identifiable*). However, assigning nodes as monitors usually incurs non-negligible operational cost (e.g., hardware/software, human efforts). Therefore, a key problem of network tomography is the *monitor assignment problem*, i.e., selecting the minimum number of nodes as monitors to identify the link metrics using measurement paths conducted between pairs of these monitors.

*This work is supported by the National Basic Research Program of China (No. 2015CB352400), National Science Foundation of China (No. 61502417, No. 61772465, and No. 61472360), Zhejiang Provincial Natural Science Foundation of China (No. LY16F020006), Zhejiang Provincial Key Research and Development Program (No. 2017C02044), and the Fundamental Research Funds for the Central Universities (No. 2017FZA5013). Yi Gao is the corresponding author.

This monitor assignment problem has been recognized in the literature with different solutions as well as different application scenarios [2–4]. Given a *static* network with a fixed topology and a set of links which we want to monitor (i.e., *interesting links* or *preferential links*), a recent work OMA [3] is an optimal algorithm to assign a minimum number of monitors to identify these links, i.e., *preferential link tomography*. Being able to designate the interesting links flexibly (e.g., the links at critical topological location) is important, and their metrics are a great concern in many applications [5]. Otherwise, we would need to assign a substantial number of monitors to identify *all* the links, which is usually unnecessary and not cost-efficient [3]. In OMA, the network topology is assumed to be static. In many scenarios, however, the network topology can keep changing over time [6], especially in the wireless networks with channel variation or the networks with moving nodes. To solve this problem, in a recent work [4], the authors proposed an effective algorithm to assign monitors to identify *all links* in *dynamic networks*. Unfortunately, this algorithm [4] cannot perform preferential link tomography, i.e., the interesting links must always be *all* links of the network.

In this paper, we focus on the problem of *preferential link tomography in dynamic networks*, i.e., how to assign the minimum number of monitors to identify the metrics of interesting links in networks with changing topologies? However, solving this problem has the following challenges. First, in OMA [3], it first partitions the static network topology (i.e., a graph) into many parts (i.e., graph components) and then sequentially assigns monitors in each part. However, with the network topology changes, this solution is no longer feasible since different network topologies cannot be partitioned into the same parts. Second, comparing with identifying *all links* in a network, assigning a minimum set of monitors to identify only the *interesting links* is much more complicated [7], which needs a deeper understanding of the dependency between the identifiability of individual links and monitors. As a result, it is challenging to *achieve* the identifiability of interesting links for a minimum monitor assignment in dynamic networks.

In order to address the above challenges, in this paper, we propose MAPLink, an efficient and effective monitor assignment algorithm to perform preferential link tomography in dynamic networks. By a careful topological analysis of the network, we first study the relationships between the identifiability of interesting links and a particular monitor. Based on this analysis, we formulate the relationships into a number of constraints on monitor assignment. Then the monitor assignment problem is converted into a constrained monitor selection optimization problem, which is a *minimum hitting set problem* and can be solved by a greedy heuristic. We further propose a constraint merging method in order to improve the performance of the algorithm. We also formally prove that the proposed algorithm is *guaranteed* to be able to identify the given sets of interesting links in the dynamic net-

978-1-5090-6501-1/17/$31.00 ©2017 IEEE

works. We implement MAPLink and evaluate its performance using two real-world dynamic networks, including a vehicular network and a sensor network. The topologies of these two networks are both changing over time due to node mobility or wireless dynamics. Compared with three baseline approaches, MAPLink achieves much better performance in terms of the number of assigned monitors.

Our contributions of this paper are summarized as follows:

- We propose MAPLink, an efficient and effective monitor assignment algorithm to perform preferential link tomography in dynamic networks for the first time.
- We analyse the efficiency of MAPLink on the monitor assignment. We also formally prove the identifiability of MAPLink, including the constraint merging method.
- We implement MAPLink and evaluate it on hundreds of realistic dynamic topologies. Our results show that MAPLink requires much fewer monitors than the baseline solutions, and our monitor assignment is highly robust against the errors in topology prediction.

The rest of this paper is structured as follows. Section II discusses the related work. Section III gives the network model and some background information on monitor assignment used in this paper. Section IV introduces the overview of our monitor assignment algorithm. Section V describes the monitor assignment algorithm in detail. Section VI presents the algorithm to improve the monitor assignment performance. Section VII theoretically proves the identifiability of MAPLink. Section VIII evaluates the proposed solutions via simulations. Finally, Section IX concludes this paper.

II. Related Work

In order to measure link metrics, different approaches have been proposed [3]. The first category includes *hop-by-hop* approaches, which use diagnostic tools such as *traceroute*, *pathchar* [8], and *Network Characterization Service (NCS)* [9] 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, i.e., network tomography. These approaches use the path information to calculate link metrics, reducing the number of probes significantly. The basic idea is to build a linear system from the path measurements and use the theory of linear algebra to calculate the unknown variables (i.e., link metrics) [10, 11]. Chen *et al*. [10] show that it is challenging to uniquely identify all link metrics due to the presence of linearly dependent paths. Later, many approaches have been proposed to optimize the selection of measurement paths. Zheng *et al*. [12] address the problem of selecting the minimum number of probing paths that can uniquely determine all the solutions for a set of target links. Gopalan *et al*. [13] develop an algorithm to determine the maximum number of linearly independent cycles and the corresponding identifiable links with a single monitor. Furthermore, many robust network tomography approaches are proposed to be resilient to the network failures. Tati *et al*. [14] study the problem of selecting paths to improve the performance of network tomography applications under failures. Dong *et al*. [15] aim at setting up topological conditions to ensure identifiability of additive link metrics in the presence of any number of link failures. Since routing along cycles is typically prohibited in real networks, cycle-free measurement paths are usually preferred.

Ma *et al*. give the necessary and sufficient conditions [2] for network identifiability and associated algorithms [16] for identifying link metrics, employing only cycle-free measurements. There are also related algorithms on achieving maximal identifiability with a limited number of monitors, or employing a minimal number of monitors to identify only a subset of links. Ma *et al*. [17] propose efficient algorithms to maximize the number of identifiable links by placing a given number of monitors, using cycle-free measurement paths. In contrast, Gao *et al*. [3] study the problem of preferential link tomography where only a subset of interesting links need to be identified. They develop scalable algorithms to place a minimum number of monitors to identify the interesting links. All the above works assume static network topology or routing with a few exceptions. He *et al*. [4] propose efficient monitor placement algorithms to identify *all* link metrics in dynamic networks. However, identifying all link metrics is usually not necessary and could require a large number of monitors, causing high overhead. Differently from these works, we study the *preferential* link tomography problem in dynamic networks.

III. Network Model and Background on Monitor Assignment

A. Models and Assumptions

Consider a fixed set of nodes (or vertices) V which form a network with time-varying topologies. We assume that the network topology can be predicted (with a certain accuracy) using existing topology prediction models such as [18, 19]. For example, it is easy to predict the temporal patterns of topology for a dynamic network formed by either public buses or satellites which have fixed tours and schedules; in low-duty-cycle sensor networks, the sleep/wake-up pattern is periodic and can be predicted accurately. In the evaluation section, we will also evaluate the performance of the monitor assignment algorithm under topology prediction errors. Based on the prediction, these time-varying topologies are represented by a sequence of undirected graphs $\{\mathcal{G}_t = (V, L_t) : t = 1, ..., T\}$, where V and L_t are the sets of nodes and links of the t-th topology, respectively. Although we focus on link changes in this paper, node changes can also be handled by modeling them as special link changes. Without loss of generality, we can assume that graph $\mathcal{G}_t$ is connected, as different connected components of the graph can be monitored separately. We denote the link incident to nodes u and v by uv, and assume that each link is symmetric, i.e., link uv and link vu have the same metric (e.g., packet loss ratio), as monitors can measure the metrics in its two directions. Given a topology $\mathcal{G} = (V, L)$ (subscript t omitted), a set $\mathcal{I} \subseteq L(\mathcal{G})$ is a set of interesting links whose metrics are interested to be identified. A subset of nodes in V are assigned as monitors and can initiate/collect end-to-end measurements to identify the link metrics in $\mathcal{I}$. Since routing along cycles is typically prohibited in real operational networks [2, 3], we only use cycle-free measurement paths (i.e., simple paths) between pairs of monitors.

A monitor m_A can initiate a measurement packet to another monitor m_B through a simple path which does not contain repeated nodes. We assume that the routing path of

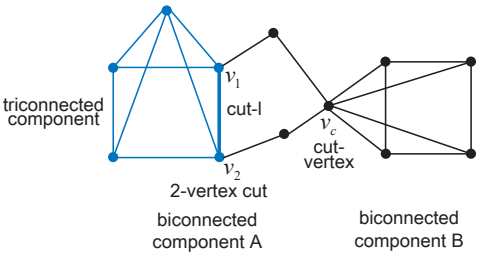

Fig. 1. A sample graph to illustrate some graph theory concepts used in this paper. The cut-vertex v_c separates the graph into two biconnected components. The 2-vertex cut $\{v_1, v_2\}$ separates one biconnected component into one triconnected component and a cycle. The link cut-l connects the two vertices in the 2-vertex cut.

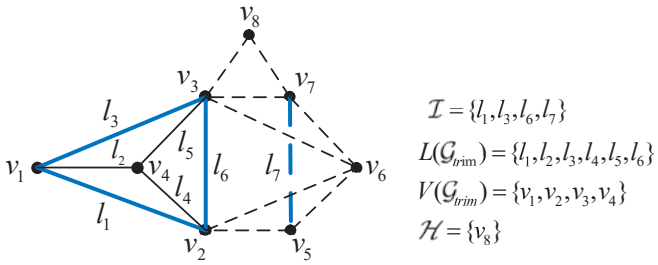

Fig. 2. An example that illustrates the output of graph trimming [7], which is a trimmed graph $\mathcal{G}_{trim}$ (solid part in this example) and a set $\mathcal{H}$ of helper vertices.

each measurement packet is *controllable* due to the following reasons. First, such controllable routing is generally supported in common networks like overlay networks and single-ISP networks [2]. In these networks, the controllable measurements can be achieved by enabling source routing (RFC791 [20]) at the routers/switches. Second, a recent work XPath [21] 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.

Monitor m_B can obtain the metric of the path measurement, which is the sum of all link metrics along the path since the link metrics are additive. Then we can build a system of linear equations, where the unknown variables are the link metrics and the known constants are the end-to-end path measurements. Thus, preferential link tomography essentially solves this linear system of equations. If the metric of an interesting link can be uniquely determined from the linear system obtained by a monitor assignment, this interesting link is *identifiable* by the monitor assignment.

B. Concepts and Definitions

The following concepts [3] 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 *biconnected* when the graph (or component) is still connected after removing any one vertex and the links incident to it. It includes at least two vertices. Note that partitioning a connected graph into biconnected components is a classical graph theory problem which can be solved in linear time [22].
- A graph (or component) is *triconnected* when the graph (or component) 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 called SPQR-tree [23], which can partition a biconnected graph into a number of SPQR components (triconnected components and/or cycles in this paper) in linear time.
- A *cut-vertex* for a connected graph $\mathcal{G}$ is a vertex whose removal will disconnect the graph $\mathcal{G}$. A *2-vertex cut* for a connected graph $\mathcal{G}$ is a set of two vertices $\{v_1, v_2\}$ such that removing v_1 or v_2 alone does not disconnect $\mathcal{G}$, but removing both disconnects $\mathcal{G}$. A *separation vertex* is a vertex that is a cut-vertex or part of a 2-vertex cut. A *cut-l* is a link that connects the two vertices in a 2-vertex cut.

Figure 1 shows an example with two biconnected components separated by a cut-vertex (i.e., v_c). The left biconnected component A is further partitioned into one triconnected component and one cycle by a 2-vertex cut (i.e., $\{v_1, v_2\}$). The link that connects the two vertices v_1 and v_2 in the 2-vertex cut is a cut-l (i.e., cut link).

C. Background on Graph Trimming

The graph trimming algorithm proposed in Scalpel [7] trims the original graph according to whether there are interesting links in each graph component. This algorithm is useful for narrowing down the search space of preferential link tomography so that it is also used in this work. In this regard, we give some related background about this graph trimming algorithm.

The input of the graph trimming algorithm is an original graph $\mathcal{G}$ and a set $\mathcal{I}$ of interesting links. The algorithm consists of the following two stages [3, 7]. The first stage trims a number of biconnected components with no interesting links, and the second stage trims a number of SPQR components (i.e., triconnected components or cycles). During the second stage graph trimming, a *helper vertex* is chosen for each SPQR component been trimmed. Specifically, in each SPQR component, we will choose the vertex v as a helper vertex if v is not incident to any interesting link and the cut-l (e.g., cut-l in Figure 1). Otherwise, any vertex (not the separation vertex) in the component will be chosen as the helper vertex. 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 which separates the trimmed graph (i.e., the remaining graph after trimming) 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 $\mathcal{G}_{trim}$ 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 $\mathcal{G}_{trim}$. Therefore, reserving these helper vertices keeps the possibility of finding a better solution after graph trimming.

Figure 2 gives an example. The dotted part of graph $\mathcal{G}$ is trimmed and the solid part is the trimmed graph $\mathcal{G}_{trim}$. Vertex v_8 is chosen as a helper vertex in the dotted part. The link that associates with v_8 is $l_6 = h^{-1}(v_8)$. Moreover, it can be

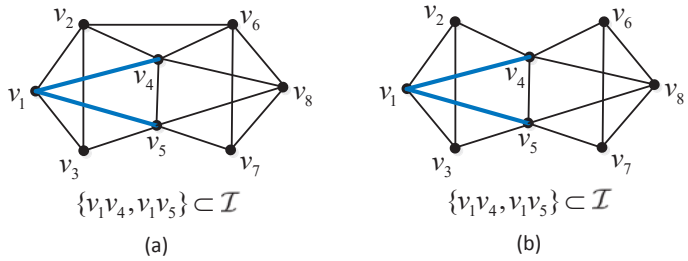

Fig. 3. A motivating example. In order to identify the two interesting links (v_1v_4, v_1v_5) in two topologies, the straightforward method will select two redundant monitors.

verified that both of the monitor assignments $\{v_4, v_1, v_2\}$ and $\{v_4, v_8\}$ are able to identify all interesting links (i.e., l_1,l_3,l_6,l_7) in $\mathcal{G}$, so we can reduce the number of monitors by using the helper vertex (i.e., v_8).

D. Motivating Example

OMA [3] is an optimal monitor assignment algorithm for preferential link tomography in a *static* network with a fixed topology. One can directly apply OMA to design an algorithm to assign monitors for a dynamic network with time-varying topologies $\{\mathcal{G}_t : t = 1, ..., T\}$, i.e., taking the union of monitors assigned in each topology $\mathcal{G}_t$ as the overall monitor assignment. This straightforward solution, however, may select many redundant monitors. For example, Figure 3 shows a dynamic network with two topologies $\mathcal{G}_1$ and $\mathcal{G}_2$. The difference of the two topologies is that the first topology $\mathcal{G}_1$ includes one more link v_2v_6. We assume that the interesting links of these two topologies are the same, i.e., $\{v_1v_4, v_1v_5\} \subset \mathcal{I}$.

$\mathcal{G}_1$ in Figure 3(a) is a triconnected graph. According to existing results [2, 3], selecting any two vertices in $\{v_2, v_3, v_6, v_7, v_8\}$ as monitors can identify the interesting links v_1v_4 and v_1v_5. Therefore, one optimal monitor assignment for $\mathcal{G}_1$ is $\{v_6, v_7\}$. Differently, for $\mathcal{G}_2$ in Figure 3(b), the topology becomes a biconnected graph and contains two triconnected components which are separated by the 2-vertex cut $\{v_4, v_5\}$. In $\mathcal{G}_2$, one optimal monitor assignment is $\{v_2, v_3\}$. Hence, applying OMA separately to $\mathcal{G}_1$ and $\mathcal{G}_2$ (i.e., the straightforward method) may generate the following monitor assignments: $M_1 = \{v_6, v_7\}$ for $\mathcal{G}_1$, and $M_2 = \{v_2, v_3\}$ for $\mathcal{G}_2$, i.e., $|M| = \left|\bigcup_{t=1}^{2} M_t\right| = 4$. Nevertheless, M_2 itself can identify both $\mathcal{G}_1$ and $\mathcal{G}_2$, i.e., two redundant monitors are selected by the straightforward method. This example illustrates the need to jointly consider all network topologies for a better monitor assignment. In this paper, we propose an efficient solution to assign much fewer monitors to simultaneously achieve interesting link identifiability for all topologies $\mathcal{G}_1$,...,$\mathcal{G}_T$.

IV. Overview of Monitor Assignment

The basic idea of our solution is to encode the monitor assignment requirement for each graph component in each topology into a number of constraints. Then by solving an optimization problem formed by these constraints, we can obtain a monitor assignment which satisfies all constraints and guarantees the identifiability of all interesting links in all topologies. More concretely, we record these constraints on monitor assignment in the form of *set-integer* pairs $F = \{(S_i, k_i) \mid i = 1, 2, ...\}$, where $S_i \subseteq V$ is a set of candidate nodes, and k_i is the minimum number of monitors which should be selected from this set.

Algorithm 1 Monitor Assignment For Dynamic Networks

Input: The trimmed graph set $\mathcal{G}_{trim}^{(1)}, ..., \mathcal{G}_{trim}^{(T)}$, the helper vertex set $\mathcal{H}^{(1)}, ..., \mathcal{H}^{(T)}$, and the interesting link set $\mathcal{I}^{(1)}, ..., \mathcal{I}^{(T)}$
Output: A monitor assignment M^S for all network topologies
1: let F be an empty set of monitor assignment constraints;
2: **for** each trimmed graph $\mathcal{G}_{trim}^{(t)}$ **do**
3: partition $\mathcal{G}_{trim}^{(t)}$ into biconnected components $\mathcal{B}_1$,$\mathcal{B}_2$,...;
4: **for** each biconnected component $\mathcal{B}_i$ with $|\mathcal{B}_i| \geq 3$ **do**
5: partition $\mathcal{B}_i$ into SPQR components $\mathcal{R}_1$,$\mathcal{R}_2$,...;
6: **for** each SPQR component $\mathcal{R}_j$ **do**
7: MCE($\mathcal{R}_j, \mathcal{H}^{(t)}, \mathcal{I}^{(t)}, F$);
8: MCE-Bi($\mathcal{G}_{trim}^{(t)}, \mathcal{H}^{(t)}, \mathcal{I}^{(t)}, F$);
9: use the extracted constraints F to form an optimization problem.
10: obtain M^S by solving the optimization problem.

Algorithm 1 gives the overview of the monitor assignment algorithm for dynamic networks. The input is the trimmed graph set $\mathcal{G}_{trim}^{(1)}, ..., \mathcal{G}_{trim}^{(T)}$ for all network topologies after the graph trimming [7], the associated helper vertex set $\mathcal{H}^{(1)}, ..., \mathcal{H}^{(T)}$ preserved in the graph trimming, and the interesting link set $\mathcal{I}^{(1)}, ..., \mathcal{I}^{(T)}$ ($\mathcal{I}^{(t)} \neq \emptyset$). The output is a monitor assignment M^S such that assigning nodes in set M^S as monitors can achieve interesting the link identifiability for all topologies. After initializing the monitor constraint set F (line 1), the algorithm partitions each $\mathcal{G}_{trim}^{(t)}$ into biconnected components and partitions each biconnected component that contains three or more nodes into SPQR components (lines 3 and 5) using existing graph partitioning algorithms [22, 23]. Then the algorithm sequentially extracts constraints for each SPQR component (line 4 to 7) by an efficient algorithm called MCE (Monitor Constraint Extraction); see Section V. After that, there may be some boundary cases about the biconnected components need to be handled with (e.g., the $\mathcal{G}_{trim}^{(t)}$ is just a single interesting link). Therefore, an algorithm MCE-Bi(.) is used to handle these boundary cases (line 8). Our strategy to obtain an optimal (or near-optimal) monitor assignment for all topologies is to first characterize the constraints on monitor assignment for each topology, and then select monitors to satisfy these constraints (lines 9 and 10). This is achieved by solving the following optimization problem:

$$\begin{aligned} \text{minimize:} \quad & |M| \\ \text{subject to:} \quad & |M \cap S_i| \geq k_i \quad \forall (S_i, k_i) \in F, M \subseteq V \end{aligned} \tag{1}$$

where $|M|$ is the number of monitors in assignment M, and V is the node set of all topologies .

In the following, we will formally present the MCE (Monitor Constraint Extraction) algorithm (Section V). Then we will introduce the complete solution to the monitor assignment in dynamic networks, including a Constraint Merging Method (CMM) which further improves the performance of the proposed algorithm (Section VI).

V. Constraint Extraction for Monitor Assignment

In this section, we will describe the MCE algorithm in detail. The input of the MCE algorithm is an SPQR component $\mathcal{R}_j$ (triconnected component or cycle), a helper vertex set $\mathcal{H}^{(t)}$, an interesting link set $\mathcal{I}^{(t)}$, and the previous set of constraints F for monitor assignment. The output is the updated set of constraints F for monitor assignment.

Algorithm 2 Monitor Constraint Extraction (MCE)

Input: An SPQR component $\mathcal{R}_j$, a helper vertex set $\mathcal{H}^{(t)}$, an interesting link set $\mathcal{I}^{(t)}$, and the previous set of constraints F
Output: The updated monitor assignment constraints F

1: **if** $\mathcal{R}_j$ is a triconnected component **then**
2: Let $V_1 = \{v | L(v) \cap \mathcal{I}^{(t)} = \emptyset, v \in V(\mathcal{R}_j)\}$
3: Let $V_2 = \{v | L(v) \cap \mathcal{I}^{(t)} = \emptyset, h^{-1}(v) \in L(\mathcal{R}_j)\}$
4: Let $V_1' = V_1 \backslash P(\mathcal{R}_j)$
5: **if** $|P(\mathcal{R}_j)| = 0$ **then**
6: **if** $|V_1| \geq 2$ **then**
7: $F \leftarrow F \cup \{(V_1, 2)\}$;
8: **else if** $|V_2| \geq 2$ **then**
9: $F \leftarrow F \cup \{(V_2, 2)\}$;
10: **else**
11: $F \leftarrow F \cup \{(V(\mathcal{R}_j), 3)\}$;
12: **else if** $|P(\mathcal{R}_j)| = 1$ **then**
13: Let $L_s = L(P(\mathcal{R}_j)) \cap \mathcal{I}^{(t)} \cap L(\mathcal{R}_j)$
14: **if** $|L_s| = 0$ and $|V_2| > 0$ **then**
15: $F \leftarrow F \cup \{(V_2, 1)\}$;
16: **else if** $|L_s| = 0$ and $|V_1'| > 0$ **then**
17: $F \leftarrow F \cup \{(V_1', 1)\}$;
18: **else if** $|L_s| = 1$ and $|\mathcal{I}^{(t)} \cap L(\mathcal{R}_j)| = 1$ **then**
19: Let $v = V(L_s) \backslash P(\mathcal{R}_j)$
20: $F \leftarrow F \cup \{(\{v\}, 1)\}$;
21: **else**
22: $F \leftarrow F \cup \{((V(\mathcal{R}_j) \backslash P(\mathcal{R}_j)), 2)\}$;
23: **else if** $|P(\mathcal{R}_j)| = 2$ **then**
24: **if** $|V_1'| > 0$ **then**
25: $F \leftarrow F \cup \{(V_1', 1)\}$;
26: **else if** $|V_2| > 0$ **then**
27: $F \leftarrow F \cup \{(V_2, 1)\}$;
28: **else**
29: $F \leftarrow F \cup \{((V(\mathcal{R}_j) \backslash P(\mathcal{R}_j)), 1)\}$;
30: **else**
31: **for** each $l \in \mathcal{I}^{(t)}$ ($l = v_1 v_2$) in the cycle **do**
32: Let l_1'(or l_2') be another link (not l) that is incident to v_1(or v_2)
33: **if** $v_1 \notin P(\mathcal{R}_j)$ **then**
34: $F \leftarrow F \cup \{(h(l_1') \cup \{v_1\} \cup \{h(l)\}, 1)\}$;
35: **if** $v_2 \notin P(\mathcal{R}_j)$ **then**
36: $F \leftarrow F \cup \{(h(l_2') \cup \{v_2\} \cup \{h(l)\}, 1)\}$;

Algorithm 2 gives the MCE algorithm. Here, $V(\mathcal{R}_j)$ denotes all the vertices in component $\mathcal{R}_j$, $L(\mathcal{R}_j)$ denotes all the links in $\mathcal{R}_j$, $P(\mathcal{R}_j)$ denotes the set of separation vertices in $\mathcal{R}_j$ (which can be computed during the graph partitioning process), and $L(v)$ denotes the set of links incident to vertex v. As mentioned in Section IV, MCE processes each triconnected component and cycle separately (lines 1 and 31 of Algorithm 2).

For each triconnected component $\mathcal{R}_j$ (line 1 to 29), three vertex sets V_1, V_2 and V_1' are introduced (line 2 to 4). In V_1, each vertex is in $\mathcal{R}_j$ and is not an endpoint of any interesting link. In V_2, each vertex is a helper vertex whose associated link is in $\mathcal{R}_j$ and is not an endpoint of any interesting link. In V_1', each vertex is in V_1 and is not the separation vertices. Then we consider three different cases (lines 5, 12 and 23) according to the number of separation vertices in $\mathcal{R}_j$.

- If $\mathcal{R}_j$ has no separation vertex (line 5 to 11), then the trimmed graph $\mathcal{G}_{trim}^{(t)}$ must be a triconnected graph. Two monitors are sufficient to identify the interesting links if there exist at least two vertices in V_1 (or V_2). In this case, the MCE algorithm will add a constraint $(V_1, 2)$ (or $(V_2, 2)$) to the set of constraints F. Figure 4(a) shows an example of this case where the constraint $(\{v_2, v_7\}, 2)$ is added to F. Otherwise, a constraint with three monitors will be added to F (line 11).
- If $\mathcal{R}_j$ has one separation vertex (line 12 to 22), then there is one cut-vertex v_c in $\mathcal{R}_j$ (i.e., $P(\mathcal{R}_j) = \{v_c\}$). We first define a link set L_s as all interesting links in $\mathcal{R}_j$ and incident to v_c (line 13). Then a constraint is extracted in four different scenarios. 1) If L_s does not include any link and there is at least one vertex in V_2, MCE adds a constraint $(V_2, 1)$ to F (line 15). 2) If there is no vertex in V_2, MCE extracts a constraint $(V_1', 1)$ if V_1' has one or more vertices (line 17). Figure 4(b) shows this case where the constraint $(\{v_1, v_3\}, 1)$ is added. 3) If L_s includes a single link and there is no other interesting link in $\mathcal{R}_j$, a constraint with another endpoint v of this interesting link will be added to F (line 20). 4) Otherwise, one monitor cannot identify all interesting links in $\mathcal{R}_j$. Therefore, a constraint with two monitors (not v_c) is added to F (line 22).
- If $\mathcal{R}_j$ has two separation vertices (line 23 to 29), then one monitor in $\mathcal{R}_j$ is sufficient to identify the interesting links if there is at least one vertex in V_1' (or V_2) and the constraint $(V_1', 1)$ (or $(V_2, 1)$) is added. Figure 4(c) shows this case where constraint $(\{v_2\}, 1)$ is added to F. Otherwise, a constraint with one monitor in $\mathcal{R}_j$ (not the separation vertices) is added to F (line 29).

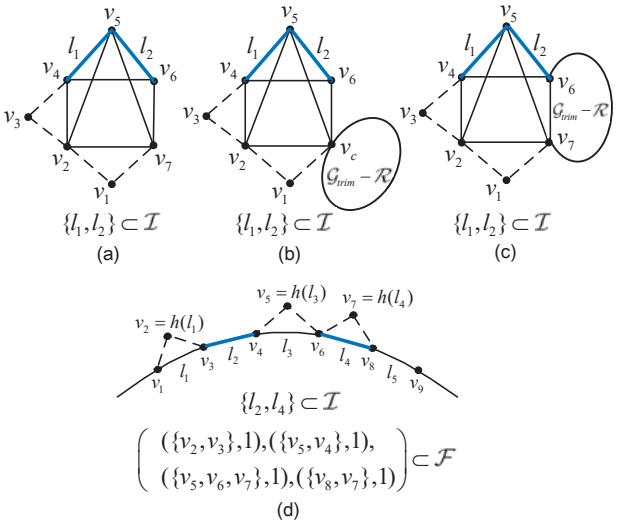

Fig. 4. Example of constraints generation in an SPQR component.

For each cycle $\mathcal{R}_j$ (line 31 to 36), MCE extracts constraints for each interesting link ($l = v_1 v_2$) in this cycle. For each vertex v_i of the two endpoints of l, if v_i is not a separation vertex, then MCE generates a constraint at v_i's side for identifying l, i.e., one vertex in $\{h(l'), v_i, h(l)\}$ should be assigned as a monitor (l' is a neighboring link which is incident to v_i). Note that if one (or two) of the three candidates does not exist, the constraint becomes that one of the rest two (or one) candidates should be assigned as a monitor. In Figure 4(d), for example, a constraint $(\{v_2, v_3\}, 1)$ is generated by MCE to identify link l_2.

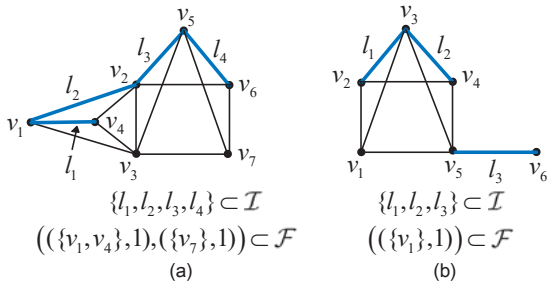

Fig. 5. Example of handling boundary cases of biconnected components.

Moreover, as shown in the sketch of the monitor assignment (Algorithm 1), there might be some boundary cases need to be solved by algorithm MCE-Bi(.) separately. Specifically, based on the previous monitor assignment (constrained by Algorithm 2), MCE-Bi(.) then checks whether further monitor assignment is required using an existing algorithm [24], and adds additional monitor constraints if necessary. For example, if the constraints in set F only record the assignment requirements for two monitors, MCE-Bi(.) needs to check whether all possible assignments of these two monitors can identify all interesting links in the graph. If not all interesting links are identifiable, an additional constraint with at least one monitor will be added to F. Figure 5(a) shows this case where two constraints have been extracted by MCE and l_1 is not identifiable. Then MCE-Bi(.) appends a constraint $(\{v_2, v_3, v_5, v_6\}, 1)$ to F, in order to identify all interesting links. Furthermore, if there is just a single interesting link in the biconnected component, F does not contain any monitor constraint for this link since MCE only handles the triconnected components and circles. Then MCE-Bi(.) first extracts a monitor constraint with an endpoint (not the separation vertices) of that link, and checks the identifiability such that additional constraints may be extracted. In Figure 5(b), a constraint $(\{v_6\}, 1)$ will be first added to F, then constraint $(\{v_2, v_3, v_4, v_5\}, 1)$ is also extracted to identify the interesting link l_3.

VI. Monitor Assignment

In this section, we study the complexity of monitor assignment and related algorithm for monitor selection in dynamic networks. Moreover, we develop an efficient method to improve the monitor assignment performance.

A. Constrained Monitor Selection

The goal of our work is to assign a set of monitors to achieve interesting link identifiability for all network topologies. Therefore, after obtaining the monitor constraints F by applying MCE to each topology $\mathcal{G}_t$ (t = 1,...,T), we formulate the problem to be selecting a minimum subset $M \subseteq V$ such that $|M \cap S_i| \geq k_i$ for all $(S_i, k_i) \in F$ (Eq. (1) in Section IV). We refer to this problem as the *minimum hitting set problem (min-HSP)* with input (V, F), as it contains the classic hitting set problem (HSP) as a special case (when $k_i \equiv 1$). Since the constraints computed by MCE are sufficient for achieving the interesting link identifiability (Theorem VII.10), we can obtain a monitor assignment that identifies the interesting links in all topologies by solving the corresponding min-HSP.

Unfortunately, min-HSP is NP-hard since HSP is NP-hard. In fact, the problem of constrained monitor selection is a special case of min-HSP under the given monitor constraints F. The following theorem shows that this problem is NP-hard. The complete proof is provided in the technical report [25] of this work.

Theorem VI.1. *The optimal monitor assignment for arbitrary topologies $\mathcal{G}_t$ ($t = 1, ..., T$) is NP-hard.*

In our problem, we can speed up the monitor computation by first determining vertices that must be monitors and excluding them from subsequent monitor selections. More specifically, let V' denote these vertices, captured by constraints $F' = \{(\{v\}, 1) : v \in V'\}$. It thus suffices to focus on selecting the remaining monitors by solving a reduced min-HSP with input $(V \backslash V', F \backslash F')$. We use a greedy heuristic for the rest monitor selections. The core of the greedy heuristic is described as follows. While there are unsatisfied constraints, select the monitor that helps in satisfying the maximum number of unsatisfied constraints, i.e., given current monitor set M, select v as the next monitor such that v is in the maximum number of node sets among $\{S_i : (S_i, k_i) \in F, |S_i \cap M| < k_i\}$. The above process repeats itself till all monitor constraints are satisfied. The approximation ratio of this greedy heuristic is given in the following theorem. The proof is provided in the technical report [25] of this work.

Theorem VI.2. *The greedy heuristic attains (1 + log|F|) approximation for min-HSP(V, F), i.e., the number of monitors selected by the greedy heuristic is at most (1 + log|F|) times larger than the optimal solution to min-HSP.*

B. Constraint Merging

In this subsection, we further propose a *Constraint Merging Method* (CMM) to reduce the number of monitors without losing identifiability. The intuition is that if we can merge multiple constraints into one constraint, not only the computation efficiency can be improved, the number of assigned monitors could also be reduced. For example, if we can merge two constraints $(V_1, 2)$ and $(V_2, 2)$ into one constraint $(V_1 \cup V_2, 2)$, the possible solutions satisfying the merged constraint are more than that satisfying the original two constraints. Therefore, CMM will enlarge the solution space, which can reduce the number of monitors. In the following, we describe CMM in detail.

In Algorithm 2, there are two candidate vertex sets V_1 (or V_1') and V_2 for the monitor assignment in most cases. The constraints extracted from these two candidate sets can generally be merged since they are equivalent for achieving identifiability. To this end, CMM merges the two vertex sets V_1 (or V_1') and V_2 together, and then selects monitors from the merged set. For example, the original MCE algorithm selects two vertices in V_1 (or V_2) as monitors for a triconnected component with no separation vertex if V_1 (or V_2) is nonempty (lines 7 and 9 in Algorithm 2), which yields a constraint (V_1, 2) (or (V_2, 2)). Instead, CMM adds the constraint ($V_1 \cup V_2$, 2) to F. Similarly, CMM takes the union of candidate vertex sets in Algorithm 2 as the new candidate vertex set for other cases of triconnected components and cycles. Accordingly, for the same example in Figure 4, the updated constraints are $(\{v_1, v_2, v_3, v_7\}, 2)$ in Figure 4(a), $(\{v_1, v_2, v_3\}, 1)$ in Figure 4(b), and $(\{v_1, v_2, v_3\}, 1)$ in Figure 4(c), respectively.

However, assigning monitors using the merged constraints may not achieve the interesting link identifiability in some special cases. For a constraint (S_i, k_i), we have assumed that all the vertices in S_i can be selected as monitors, whereas there may exist some vertex pairs where the vertices cannot be

monitors at the same time when $k_i \geq 2$. In this regard, these vertex pairs are recorded into a test set (denoted by S^*) in the process of monitor constraint generating. To address this problem, CMM also includes an additional check procedure which is aimed to make the interesting links identifiable under our monitor assignment. More specifically, for each constraint (S_i, k_i) in F, if S_i has a subset S_i' in S^* and current assignment M (computed by the greedy heuristic) contains all of the vertices in S_i', which implies the existence of unidentifiable interesting links. Then, CMM adds one more constraint $(S_i \backslash S_i', 1)$ to F and recomputes a new assignment M, i.e., performing the back-off for monitor selection. Multiple back-offs may be needed until M contains at least one vertex in $S_i \backslash S_i'$. Hence, CMM can always guarantee the identifiability of all interesting links (see Theorem VII.9).

C. Complexity Analysis

The complexity of MAPLink mainly depends on the complexity of MCE (including CMM) and the greedy heuristic for min-HSP. First, MCE has the same complexity as OMA [3], which is $O(|V| + |L_t|)$ for each $\mathcal{G}_t$ $(t = 1, ..., T)$. Moreover, the greedy heuristic incurs $O(|V|)$ complexity for testing the satisfaction of $O(T|V|)$ constraints, resulting in a complexity of $O(T|V|^2)$. For the additional check cases, CMM repeats the back-off for $O(T|V|)$ constraints, which takes $O(T|V|^2)$ time. Therefore, the overall complexity is $O(T|V|^2)$.

VII. Theoretical Analysis

In this section, we theoretically analyze MAPLink. In particular, we will formally prove that MAPLink can *guarantee* the identifiability of *all* the interesting links in *all* topologies.

A. Existing Results

Before starting analyzing MAPLink, we first give some important results from existing works [2, 17].

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

Corollary VII.2. *If $\mathcal{T}$ is a triconnected 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) one or more vertices v_i is not in $\mathcal{T}$, but there are distinct paths (without repeated vertices) from v_i to $\mathcal{T}$.*

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

Theorem VII.3. *For a triconnected component $\mathcal{T}$ with two monitors assigned in it (or two separation vertices which connect to two monitors through two paths without repeated vertices outside $\mathcal{T}$, or one monitor in $\mathcal{T}$ and one such separation vertex), all its interior links are identifiable and all its exterior links are unidentifiable.*

B. Identifiability of MAPLink

Figure 6 shows the overview of our theoretical analysis. Three key theorems and corollary will be formally proved in this subsection, i.e., Theorem VII.9 (CMM still guarantees the identifiability), Theorem VII.10 (identifiability for each topology), and Corollary VII.11 (identifiability for all topologies).

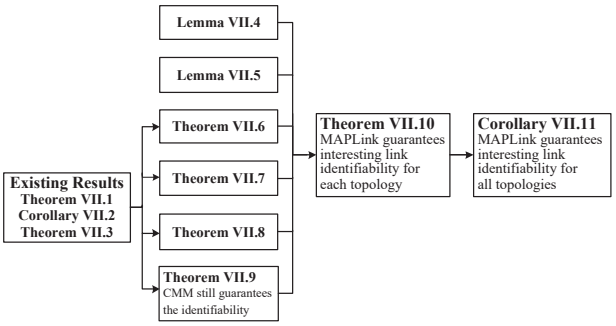

Fig. 6. Overview of theoretical analysis.

In order to prove the identifiability of MAPLink, we first introduce the following two lemmas which explicitly describe the monitor assignment relationships of MAPLink and OMA [3] for each triconnected component and each cycle, respectively.

Lemma VII.4. *For a triconnected component, there are two cases of monitor assignment M satisfying F. 1) There exists a possible assignment by OMA that is a subset of M. 2) Any possible assignment by OMA is not the subset of M.*

Lemma VII.5. *For a cycle component, there are two cases of monitor assignment M satisfying F. 1) There exists a possible assignment by OMA that is a subset of M. 2) Any possible assignment by OMA is not the subset of M.*

The full proofs of these two lemmas are provided in the technical report [25] of this work.

Note that each of the two cases can always be met for the monitor assignments. Based on the above two lemmas, we analyze the identifiability of the monitor set M assigned by MAPLink under three cases: 1) there exists a possible assignment by OMA that is a subset of M; 2) any possible assignment by OMA and M do not intersect; 3) any possible assignment by OMA partially intersects M. In the following, we will give three theorems with respect to these three different cases.

Theorem VII.6. *If a monitor assignment M satisfies the constraints F for a network topology $\mathcal{G}$, all interesting links of $\mathcal{G}$ are identifiable under M, when there exists a possible assignment by OMA that is a subset of M.*

Proof of Theorem VII.6: If the assignment by OMA is a subset of M, then there are two possible scenarios, i.e., the deterministic assignment, or these selected monitors are from the same vertex set. Clearly, in both scenarios, M is able to identify all interesting links in $\mathcal{G}$ since OMA is guaranteed to achieve identifiability of these interesting links [3]. ■

Theorem VII.7. *If a monitor assignment M satisfies the constraints F for a network topology $\mathcal{G}$, all interesting links of $\mathcal{G}$ are identifiable under M, when any possible assignment by OMA is disjoint from M.*

Proof of Theorem VII.7: According to the graph partitioning algorithm in [22], each link in $\mathcal{G}$ must be in a biconnected component $\mathcal{B}$ after partitioning graph $\mathcal{G}$. If $\mathcal{B}$ is a single interesting link, then the monitor assignment is deterministic. Therefore, M can identify this link metric according to Theorem VII.6. If $\mathcal{B}$ has more than two vertices. then $\mathcal{B}$ can be partitioned into SPQR components and each link in $\mathcal{B}$ will be

in at least one SPQR component [23]. Then, we focus on the identifiability of interesting links in each SPQR component of $\mathcal{G}$ under assignment M. Since an SPQR component (denoted by $\mathcal{R}$) can be a triconnected component or a cycle, we analyze the interesting link identifiability separately.

In algorithm 2, we introduce two vertex sets V_1, V_2 in the trimmed graph (i.e., the remaining graph after graph trimming [7]) and select monitors from these two sets. In V_1, each vertex is in $\mathcal{R}$ and is not in any interesting link. In V_2, each vertex is a helper vertex of a link in $\mathcal{R}$ and is not in any interesting link. When the assignment by OMA and MAPLink are disjoint, the monitors assigned by OMA are all in V_1 and the monitors assigned by M are all in V_2, or vice versa. According to Theorem VII.3, all links in $\mathcal{R}$ except for those incident to the monitors are identifiable. Since the vertices in V_2 (or V_1) are not in any interesting link in the triconnected component (or cycle), all interesting links in $\mathcal{R}$ are identifiable under M irrespective of whether $\mathcal{R}$ is a triconnected component or a cycle. Then we consider the identifiability of the trimmed interesting links. In the graph trimming stage [7], a cycle is trimmed only when there is no interesting link in it, so there are no interesting links in the cycles been trimmed. However, it is possible that a triconnected component with some interesting links is trimmed when these interesting links are not incident to the separation vertices. There are three possible cases of the trimmed triconnected component $\mathcal{T}'$ depending on the number of monitors in $\mathcal{T}'$.

- If $\mathcal{T}'$ contains more than one monitor, its interesting links are identifiable as these monitors are not incident to any interesting link, according to Theorem VII.3.
- If $\mathcal{T}'$ contains one monitor, there remains at least one monitor in neighboring components, otherwise none of its links can be probed via cycle-free paths. Thus, its interesting links are identifiable by using these two monitors because the interesting links are not incident to the separation vertices.
- If $\mathcal{T}'$ contains no monitors, there exist at least two monitors in neighboring components, so its interesting links are also identifiable.

The above analysis shows that the trimmed interesting links are still identifiable under M. Therefore, assignment M can identify all interesting links in $\mathcal{G}$ when any possible assignment by OMA and M are disjoint. ■

Theorem VII.8. *If a monitor assignment M satisfies the constraints F for a network topology $\mathcal{G}$, all interesting links of $\mathcal{G}$ are identifiable under M, when any possible assignment by OMA partially intersects M.*

The arguments of Theorem VII.8 are similar to the proof of Theorem VII.7. Detailed proof of Theorem VII.8 is included in the technical report [25] of this work.

In the design of MAPLink, we further use a constraint merging method (CMM) to improve the monitor assignment performance. The following theorem describes that CMM does not affect the identifiability of MAPLink.

Theorem VII.9. *Using CMM in MAPLink still guarantees the identifiability of interesting links.*

The complete proof of this theorem is given in the technical report [25] of this work.

Based on the above four fundamental theorems on the properties of M assigned by MAPLink and the guarantee of identifiability, we can show that the constraints F are sufficient for identifying all interesting links in a network topology $\mathcal{G}$.

Theorem VII.10. *The constraints extracted by MCE for an input network topology $\mathcal{G}$ are sufficient for a monitor assignment to identify its interesting links, i.e., if a monitor assignment $M \subseteq V$ satisfies F, i.e., $|M \cap S_i| \geq k_i$ for all $(S_i, k_i) \in F$, then M is able to identify all interesting links in $\mathcal{G}$.*

Proof of Theorem VII.10: If a biconnected component $\mathcal{B}$ is a single link which is an interesting link, then assignment M is deterministic, i.e, ensuring that the number of cut-vertices and monitors in $\mathcal{B}$ is two. Hence, the link is identifiable. Then we consider a biconnected component $\mathcal{B}$ with more than two vertices. In this case, $\mathcal{B}$ can be decomposed into a number of SPQR components and each link in $\mathcal{B}$ will be in at least one SPQR component. Let $\mathcal{R}_1$,...,$\mathcal{R}_n$ denote the set of SPQR components decomposed from $\mathcal{G}$. Note that each SPQR component could be a triconnected component or a cycle. According to Theorem VII.6, Theorem VII.7, and Theorem VII.8, M can identify all interesting links in all cases of monitor assignment M that satisfies F. Therefore, the constraints computed by MCE for a topology $\mathcal{G}$ are sufficient for a monitor assignment to identify $\mathcal{G}$. ■

Theorem VII.10 shows that MAPLink can achieve interesting link identifiability for each topology. Then, the following corollary formally states the interesting link identifiability for *all* topologies, which is actually equivalent to Theorem VII.10.

Corollary VII.11. *Let F be the overall set of constraints extracted by MCE for all topologies $\mathcal{G}_t$ (t = 1,...,T). Then the solution to min-HSP(V, F) yields a monitor assignment for identifying all interesting links in $\mathcal{G}_1$,...,$\mathcal{G}_T$.*

VIII. Performance Evaluation

In this section, we evaluate our algorithms using real-world dynamic networks. We first give the evaluation methodology, including the evaluation metrics and the detailed description of the network topologies used. Then we present the evaluation results.

A. Methodology

We collect real network topologies for the evaluation. We use two datasets:

1) taxi cab traces from San Francisco [26], from which we select traces of 100 nodes over a 4-hour period with location updates roughly every minute. We extract dynamic topologies from each trace by specifying a communication range and connecting two nodes by a link whenever they are within the range. See Table 1 for a summary of extracted topologies under different ranges.

Table 1: Dynamic Topologies For Taxi Network (100 Nodes)

range (m)	#topology changes	avg #links	avg #components
500	240	179.6	12.8
1000	240	542.0	8.6
1500	240	1032.2	4.8
2000	240	1546.3	3.6
2500	240	2023.1	2.7
3000	240	2446.9	1.8
3500	240	2803.9	1.2

2) dynamic network topologies generated by the GreenOrbs project [27] which is a real WSN application for forest surveillance from 2008. About 400 TelosB nodes are scattered

Table 2: Dynamic Topologies For GreenOrbs (330 Nodes)

time slot (h)	#topology changes	avg #links	avg #components
1.5	112	1121.1	18.4
2	84	2516.9	11.6
3	56	3277.6	7.7
4	42	4198.1	3.8
8	21	5187.5	2.6
16	10	6054.9	1.8

on the Tianmu Mountain, China and gather temperature, light, humidity, and carbon absorbance once every 10 minutes. The data set used for evaluation in this paper mainly comes from the operational period of GreenOrbs in December 2010. It contains data of 7 consecutive days, counts 385488 data packets. We extract dynamic topologies from each trace by specifying a time slot and connecting two nodes by a link if there is at least one packet transmitted between these two nodes within the slot. The GreenOrbs project uses the collection tree protocol [28] as the routing protocol, in which a node dynamically selects a *parent node* to forward all its packets to it. Note that a sensor node sometimes switches its parent node due to wireless dynamics, resulting in dynamic routing paths to the sink node. Table 2 shows the detailed information of the extracted topologies.

As expected, the network becomes denser (with a larger number of links) and better connected (with a smaller number of connected components) as the range and the span of slot increase.

The evaluation metrics are the number of assigned monitors and the execution time. We compare the following algorithms:

- The incremental MMP algorithm (IMMP) [4]. This state-of-the-art algorithm incrementally applies MMP to each topology. Based on the existing monitors, it assigns the minimum number of additional monitors to identify the subsequent topologies. Note that MMP is an extreme case of the preferential link tomography problem where all links are interesting links.
- The union-OMA algorithm (uOMA). This algorithm sequentially applies OMA to compute a monitor assignment for each $\mathcal{G}_t(t = 1, ..., T)$, and then takes the union as the overall monitor assignment.
- The basic-MAPLink algorithm (bMAPLink). This algorithm is the basic version of MAPLink which does not use CMM to merge the monitor constraints generated by MCE.
- The MAPLink. This is the proposed algorithm in this paper. It first applies MCE to extract the monitor constraints, and then merges the monitor constraints by CMM. A greedy heuristic is used to solve the associated min-HSP for the resultant monitor assignment.

For each network topology, we randomly choose different number of links as interesting links: 5% and 50% of all links in the original topology. We run each simulation 10 times and report the average results for our algorithms. Since MMP always views all links as interesting links, we apply IMMP once for the topologies under the same range and slot.

B. Results

1) Monitor Assignment: We first compare different monitor assignment algorithms in terms of the number of monitors assigned (note that all algorithms guarantee the interesting link identifiability). Figure 7 shows the monitor assignment

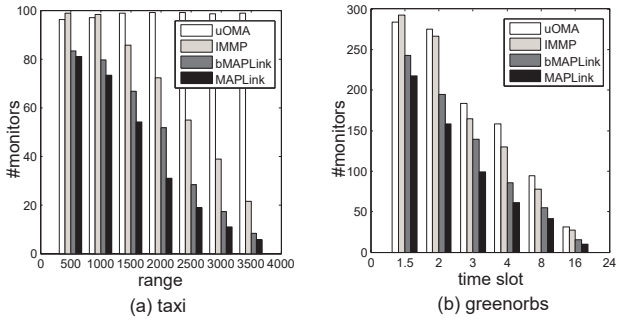

Fig. 7. Comparing different assignment algorithms when the interesting link rate is 5%.

results for the taxi network and greenorbs network when the interesting link rate is 5%. We observe that IMMP consistently assigns a substantial number of unnecessary monitors to identify the specified interesting link metrics over all topologies. As expected, the uOMA algorithm uses many more monitors than bMAPLink and MAPLink. Although bMAPLink works well in theses traces, we see that it can be further improved by the MAPLink algorithm, owing to the monitor constraint merging. The results also show that different topologies require different number of monitors to identify the same number of interesting links. For example, using MAPLink to assign monitors in the taxi network, 80 and 5 monitors are needed when communication ranges are 500 metres and 3500 metres, respectively. The reason is that a dense network can enable more measurement paths to identify the interesting links, and it usually requires fewer monitors than a sparse network.

Figure 8 summarizes the assignment results when the interesting link rate is 50%. It is also worth noting that IMMP still requires more monitors to identify the interesting link metrics. Similarly, MAPLink continues to outperform uOMA and bMAPLink. Comparing the results with Figure 7, more monitors are assigned since there are more interesting links.

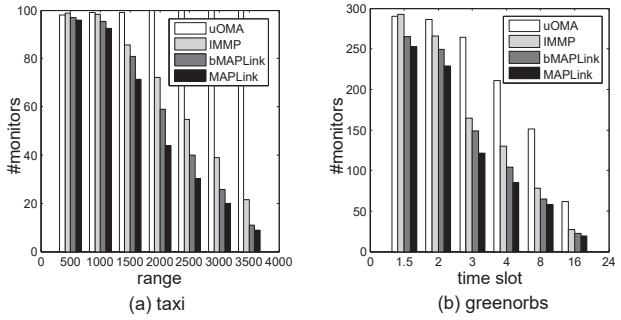

Fig. 8. Comparing different assignment algorithms when the interesting link rate is 50%.

Moreover, we also evaluate the overhead of different monitor placement algorithms using the taxi network topologies. Table 3 compares the computation time of the proposed algorithms to finish the monitor placement when 50% of the links are interesting links. We perform the simulations on a computer with Inter(R) Core i3-2100 CPU @ 3.1GHz, 4.0 GB memory, and 64-bit Win7 OS. As we can see, IMMP is much faster due to lack of the graph trimming, the monitor constraint extraction and the constrained monitor selection process. In most traces, uOMA and bMAPLink require comparable com-

Table 3: Computation Time (seconds, interesting link rate = 50%)

range (m)	IMMP	uOMA	bMAPLink	MAPLink
500	9.35	25.43	38.56	39.79
1000	38.29	100.58	141.54	144.08
1500	129.12	365.34	496.22	528.70
2000	256.02	744.42	1007.69	1312.52
2500	351.87	951.51	1268.92	1734.94
3000	560.24	1524.39	2036.79	2392.58
3500	822.25	2216.60	3275.98	3402.23

putation time. Although the computation time is longer, it is feasible to perform the monitor assignment by MAPLink in various networks.

2) Impact of Prediction Error: So far we have assumed accurate knowledge of topologies. We further evaluate how the proposed monitor assignment algorithms perform when the input topologies contain errors. To evaluate it, we add i.i.d. zero-mean Gaussian noise with variance σ^2 to node locations of taxi traces and treat the resulting topologies as the new *ground truth topologies*. The topologies computed from the original trace are provided to the monitor assignment algorithm MAPLink as *predicted topologies*. We evaluate the robustness of the resulting assignment from two aspects: (i) testing whether the original assignment achieves the interesting link identifiability for each newly generated topology, and (ii) if not, computing the number of additional monitors needed to achieve the interesting link identifiability.

Figure 9 reports the number of increased monitors for the taxi network when the communication range is 2000 metres and the interesting link rate is 5%. As we can see, MAPLink is robust to prediction error. Specifically, MAPLink achieves identifiability for more than 95% of time, even if the error in node location is nearly 1/3 as large as the communication range. For the rest of the time, we only need to add at most three monitors to achieve the interesting link identifiability.

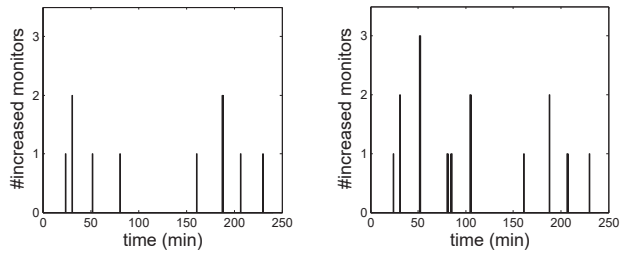

Fig. 9. Performance under prediction error when the communication range is 2000 meters and the interesting link rate is 5%.

IX. Conclusion

In this paper, we investigate the problem of assigning monitors to identify interesting link metrics from end-to-end measurements in the face of topology changes. We propose MAPLink, a monitor assignment algorithm to address this tomography problem. MAPLink first formulates the relationship between each monitor assigned and the identifiability of interesting links into a number of constraints. Then the monitor assignment problem is transformed into solving the corresponding optimization problem which is a minimum hitting set problem. We further propose a constraint merging method to improve the performance of the algorithm. We also theoretically prove the identifiability of MAPLink, including the constraint merging method. Our evaluations on realistic dynamic topologies verify the effectiveness and robustness of the proposed solution.

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, "Inferring link metrics from end-to-end path measurements: Identifiability and monitor placement," *IEEE/ACM Trans. on Networking*, vol. 22, no. 4, pp. 1351–1368, 2014.

[3] W. Dong, Y. Gao, W. Wu, J. Bu, C. Chen, and X. Y. Li, "Optimal monitor assignment for preferential link tomography in communication networks," *IEEE/ACM Trans. on Networking*, vol. 25, no. 1, pp. 210–223, 2017.

[4] T. He, L. Ma, A. Gkelias, K. K. Leung, A. Swami, and D. Towsley, "Robust monitor placement for network tomography in dynamic networks," in *Proc. of IEEE INFOCOM*, 2016.

[5] J. Wu, Y. Zhang, Z. M. Mao, and K. G. Shin, "Internet routing resilience to failures: analysis and implications," in *Proc. of ACM CoNEXT*, 2007.

[6] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro, "Time-varying graphs and dynamic networks," in *Proc. of ADHOC-NOW*, 2011.

[7] Y. Gao, W. Dong, W. Wu, C. Chen, X. Y. Li, and J. Bu, "Scalpel: Scalable preferential link tomography based on graph trimming," *IEEE/ACM Trans. on Networking*, vol. 24, no. 3, pp. 1392–1403, 2016.

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

[9] G. Jin, G. Yang, B. R. Crowley, and D. A. Agarwal, "Network characterization service (ncs)," in *Proc. of IEEE HPDC*, 2001.

[10] Y. Chen, D. Bindel, H. Song, and R. H. Katz, "An algebraic approach to practical and scalable overlay network monitoring," in *Proc. of ACM SIGCOMM*, 2004, pp. 55–66.

[11] O. Gurewitz and M. Sidi, "Estimating one-way delays from cyclic-path delay measurements," in *Proc. of IEEE INFOCOM*, 2001.

[12] Q. Zheng and G. Cao, "Minimizing probing cost and achieving identifiability in probe-based network link monitoring," *IEEE Trans. on Computers*, vol. 62, no. 3, pp. 510–523, 2013.

[13] A. Gopalan and S. Ramasubramanian, "On the maximum number of linearly independent cycles and paths in a network," *IEEE/ACM Trans. on Networking*, vol. 22, no. 5, pp. 1373–1388, 2014.

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

[15] W. Ren and W. Dong, "Robust network tomography: k-identifiability and monitor assignment," in *Proc. of IEEE INFOCOM*, 2016.

[16] L. Ma, T. He, K. K. Leung, D. Towsley, and A. Swami, "Efficient identification of additive link metrics via network tomography," in *Proc. of IEEE ICDCS*, 2013, pp. 581–590.

[17] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Monitor placement for maximal identifiability in network tomography," in *Proc. of IEEE INFOCOM*, 2014, pp. 1447–1455.

[18] M. Zhao and W. Wang, "Analyzing topology dynamics in ad hoc networks using a smooth mobility model," in *IEEE WCNC*, 2007.

[19] Y. Gu and T. He, "Dynamic switching-based data forwarding for low-duty-cycle wireless sensor networks," *IEEE Trans. on Mobile Computing*, vol. 10, no. 12, pp. 1741–1754, 2011.

[20] RFC791 (Internet Protocol). [Online]. Available: https://tools.ietf.org/html/rfc791

[21] S. Hu, K. Chen, H. Wu, W. Bai, C. Lan, H. Wang, H. Zhao, and C. Guo, "Explicit path control in commodity data centers: Design and applications," in *Proc. of USENIX NSDI*, 2015, pp. 15–28.

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

[23] G. Di Battista and R. Tamassia, "On-line graph algorithms with spqr-trees," in *Proc. of Automata, Languages and Programming*, 1990.

[24] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Link identifiability in communication networks with two monitors," in *Proc. of IEEE Globecom*, 2013, pp. 1513–1518.

[25] "Preferential link tomography in dynamic networks [technical report]," May 2017. [Online]. Available: https://www.dropbox.com/s/8jtcjhg97r2teoo/maplink-techrep.pdf

[26] Dataset of mobility traces of taxi cabs in San Francisco, USA. [Online]. Available: http://crawdad.org/epfl/mobility

[27] 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 *Proc. of ACM SenSys*, 2009, pp. 99–112.

[28] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection Tree Protocol," in *Proc. of ACM SenSys*, 2009, pp. 1–14.