Concentrated Network Tomography and Bound-based Network Tomography

by

Cuiying Feng
B.Sc., Northeast Forestry University, 2014

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

Concentrated Network Tomography and Bound-based Network Tomography

by

Cuiying Feng
B.Sc., Northeast Forestry University, 2014

Supervisory Committee

Dr. Kui Wu, Supervisor
(Department of Computer Science)

Dr. Venkatesh Srinivasan, Department Member
(Department of Computer Science)

Dr. Yang Shi, Outside Member
(Department of Mechanical Engineering)

Supervisory Committee

Dr. Kui Wu, Supervisor
(Department of Computer Science)

Dr. Venkatesh Srinivasan, Department Member
(Department of Computer Science)

Dr. Yang Shi, Outside Member
(Department of Mechanical Engineering)

ABSTRACT

Modern computer networks pose a great challenge for monitoring the network performance due to their large scale and high complexity. Directly measuring the performance of internal network elements is prohibitive due to the tremendous overhead. Alternatively, network tomography, a technique that infers the unobserved network characteristics (e.g., link delays) from a small number of measurements (e.g., end-to-end path delays), is a promising solution for monitoring the internal network state in an efficient and effective manner. This thesis initiates two variants of network tomography: concentrated network tomography and bound-based network tomography. The former is motivated by the practical needs that network operators normally concentrate on the performance of critical paths; the latter is due to the need of estimating performance bounds whenever exact performance values cannot be determined.

This thesis tackles core technical difficulties in concentrated network tomography and bound-based network tomography, including (1) the path identifiability problem and the monitor deployment strategy for identifying a set of target paths, (2) strategies for controlling the total error bound as well as the maximum error bound over all network links, and (3) methods of constructing measurement paths to obtain the tightest total error bound. We evaluate all the solutions with real-world Internet service provider (ISP) networks. The theoretical results and the algorithms developed in this thesis are directly applicable to network performance management in various types of networks, where directly measuring all links is practically impossible.

Contents

List of Tables

List of Figures

List of Abbreviations

NT	Network Tomography
MP	Measurement Path
ISP	Internet Service Provider
VNF	Virtualized Network Function
QoS	Quality of Service
SDN	Software-Defined Networking
RRT	Round Trip Time
WAN	Wide-Area Network
MPIP	Monitor Placement for Interested Paths
$\mathcal{MEB}$	Maximum Error Bound
CMMP	Constructing Minimal Measurement Paths
$\mathcal{TEB}$	Total Error Bound
ICMP	Internet Control Message Protocol
SLM	Sequentially Learning-based Measurement
OMB	One Batch Measurement
MMP	Minimum Monitor Placement
OMA	Optimal Monitor Assignment
OMP	Optimal Monitor Placement
PIP	Path Identification Problem
BC	Bi-connected Component
TC	Tri-connected Component
AS	Autonomous System
SLA	Service-Level Agreement
LSM	Linear System Model

NBI	Natural Bound Interval
TNB	Tighetest Natural Bound (Interval)
MREB	Maximally Reducing $\mathcal{TEB}$
MAIL	Maximize Additional Identifiable Link
MPMM	Minimum Monitor Placement for Maximum Reduction on $\mathcal{MEB}$
TTL	Time-To-Live

ACKNOWLEDGEMENTS

I would like to thank:

Ying Huang, Joe Winter and Yueling He, for supporting me in the low moments.

Dr. Kui Wu, for mentoring, support, encouragement, and patience.

Dr. Venkatesh Srinivasan and Dr. Yang Shi, for serving in the supervisory committee and helping me improve the thesis.

Dr. Di Niu, for helping me as the external examiner.

DEDICATION

To my favorite thing on earth that is on the Moon.

Chapter 1

Introduction

1.1 Overview

The past few decades witnessed a tremendous growth in the global network infrastructure. Modern communication networks explode not only in size of topology, but also in the inherent heterogeneity brought by inter-networks, third-party infrastructure, and large volume of transferred data loaded by diversified services (online gaming, real-time streaming). To make the network operate smoothly, it is critical to find efficient solutions for monitoring the performance of internal network links. Clearly, due to the large-scale of networks, it is infeasible to directly measure the performance of all network links.

Network slicing, another new trend in computer networks, also makes network monitoring challenging. Recent technical development on network slices allows an ISP to dynamically form different virtual networks, each for a dedicated network application [1]. A virtual network, consisting of virtualized network functions (VNFs), is a logical network tailored by the ISP to provide the corresponding service. A virtual link between VNFs may be realized as a (multi-hop) physical path. In this context, the physical resource for a virtual network is allocated on demand of the required service, and the allocated physical resource to the virtual network may change over time as long as the required Quality of Service (QoS) is satisfied. Monitoring and validating the performance of virtual networks are basic requirements for the providers of virtual networks [1]. Nevertheless, direct measuring all links' performance is generally prohibitive due to the large size of a network, the complex cooperation required from common administration, and the lack of protocol support at internal network elements.

To tackle the above difficulties, a well-known strategy is to infer the performance of internal links via end-to-end measurements. This solution is termed as network tomography [14, 31, 44, 46]. The concept of network tomography [44] was first introduced by Vardi in 1996. Since then, tremendous research efforts have been devoted to this area. In general, network tomography covers any method that infers unobserved network characteristics (e.g., link delays) by a small number of measurements (e.g., end-to-end path delays). Example network tomography problems include *inferring traffic matrix* [47, 48], *inferring network topology* [7, 37], *inferring network performance* [16, 39].

This thesis tackles one of the particular interesting case in network tomography: *inferring the*

performance of internal links by end-to-end path measurements with additive metrics. An additive metric means the value of the end-to-end path equals the sum of individual values of all links along the path. For example, delay is additive since the delay of a path equals the sum of delays of all links on the path. Packet loss rate, after a logarithmic operation, is also additive. In this specific case of network tomography, many research papers can be found [14, 28, 31, 32]. Nevertheless, no research paper has touched the two problems from industrial practice:

1. Concentrated network tomography: In most large-scale internet backbones or in the "network-as-a-service" paradigm, network operators have a strong need to know the metrics of critical paths running services to their users/tenants. In other words, their concern is mainly concentrated on the performance of critical paths rather than the performance of all links or all paths. We call this shift of focus as concentrated network tomography.

2. Bound-based network tomography: Existing research is mostly Boolean-based, i.e., whether or not an internal link is identifiable[1]. If a link is not identifiable, no further information can be provided. In many cases, however, the network operators also want to know the upper and lower bounds of a link's performance even if the link is not identifiable. We call this shift as bound-based network tomography.

This thesis is the first to raise the above new problems and pioneers the research in the above problem domains.

1.2 Research Objectives and Contributions

This thesis is targeted at solving several core problems in concentrated network tomography and bound-based network tomography. In particular,

1.2.1 Path identifiability and Monitor Placement in Concentrated Network Tomography

Substantial literature has investigated link identifiability problems [14,20,31,34], ranging from different aspects targeting at identifying metric on individual links, but the problem of path identifiability has not been touched and the network operator may have a strong need to know the performance of critical paths. Path identifiability problem is largely different from and much harder than link identifiability problem. In this regard, we have made the following contributions:

i) We study for the *first* time the necessary and sufficient topological conditions for identifying additive path metric using controllable and cycle-free measurement paths.

ii) We develop an efficient algorithm (MPIP) that requires the minimum number of monitors to identify a set of given interested paths.

iii) We evaluate our algorithm (MPIP) over real-world ISP topology. Experimental results show that compared to other link-based solutions, our solution leads to a saving of up to 40% fewer monitors.

[1] A link is called identifiable if its performance value can be uniquely determined.

1.2.2 Controlling the Total Estimation Error in Bound-based Tomography

Existing results in network tomography are mainly Boolean-based, i.e., they check whether or not a link is identifiable, and return the exact value on identifiable links. If a link is not identifiable, Boolean-based solution gives no performance result for the link. For this matter, we extend Boolean-based network tomography to bound-based network tomography where the lower and upper bounds are derived for unidentifiable links. A link's error bound equals to its upper bound minus its lower bound, and *total error bound*, computed by the sum of each link's error bound is a critical metric in bound-based tomography. With respect to this metric, we have made the following contributions:

i) We develop an efficient algorithm to obtain the *tightest* total error bound over a given network with pre-determined monitors.

ii) We give the theoretical proof showing that the total error bound obtained by our algorithm is tightest.

iii) Furthermore, we propose a method to deploy a new monitor over pre-determined monitors such that the total error bound inferred with the new monitor deployment could be maximally reduced.

iv) The evaluation results shows that, comparing with 2 benchmark monitor deployment strategies, our monitor deployment method can lead to up to 15 and 2.4 times more reduction on total error bound, respectively.

1.2.3 Controlling the Maximum Estimation Error in Bound-based Tomography

In the context of bound-based network tomography, total error bound, as one of the most important metric, has been thoroughly studied in our second work. The total error bound captures the overall error across the whole network, or in other words the average estimation error in the whole network. This type of information alone, while useful to provide the network manager with the average performance in the network, cannot be used to identify potentially congested individual links. Thereby, we aim at controlling another meaningful metric–maximum error bound. We hereby refer the link with the maximal error bound as *maximal link*. Our major contributions include:

i) With pre-determined monitors, we develop an efficient solution to minimize the maximum error bound ($\mathcal{MEB}$) over all the links with one more monitor.

ii) We theoretically prove that the newly-deployed monitor is at the "best" place, in the sense that the error bound of current *maximal link* could be maximally reduced.

1.2.4 Minimum Path Construction

With respect to constructing measurement paths, our contributions include:

i) Given a set of deployed monitors, it generally requires building all possible measurement paths (MPs) to obtain the tightest total error bound. Nevertheless, the total number of possible MPs is huge, and it is well known that listing MPs between two monitors is $\#P$-complete [43]. We then develop a path construction method (CMMP) that only uses necessary and sufficient MPs needed for obtaining the tightest total error bound.

ii) CMMP constructs the minimum number of MPs on the premise that all the MPs must be designed before measurement process, thus zero knowledge is provided regard the measurement data. If the premise could be relaxed to allow sequential measurement, i.e., partial knowledge of the measurement data will be revealed gradually along with the selection of MPs, we show that the number of MPs could be further reduced.

1.3 Thesis Outline

The rest of this thesis is organized as follow. In Chapter 2, we provide the most relevant related work in the literature of network tomography and give the problem formulation universally applicable to each work in this thesis. Chapter 3 solves the path identifiability problem and the optimal monitor placement problem in concentrated network tomography. Chapter 4 extends Boolean-based tomography to bound-based tomography, under which a significant metric (i.e., total error bound $\mathcal{TEB}$) is thoroughly studied and several results are shown thereby. Conducting further study on bound-based network tomography, we propose a method to control another meaningful metric (maximal error bound) in Chapter 5. In Chapter 6, we propose two optimal measurement path construction algorithms, in the sense that the number of MPs needed for obtaining $\mathcal{TEB}$ is reduced to the utmost extent. Finally, Chapter 7 concludes this thesis and presents future work.

Chapter 2

Related Work and Problem Formulation in Network Tomography

2.1 Related Work

The concept of network tomography was first introduced in [44]. Since then, network tomography has been extensively investigated, with the three main goals: *inferring traffic matrix* [36,47,48], *inferring network topology* [7, 17, 37] and *inferring network performance* [4–6, 12, 16, 18, 21, 26, 27, 29, 39].

Most existing work in network performance tomography targeted at the identifiability problem for additive metrics. The objects in study are either identifying all links [31] (*Complete Identifiability*), or identifying preferential links [14, 20] (*Partial Identifiability*). Studying whether or not the objects are identifiable[1] with existing monitors (i.e., nodes capable of sending and collecting measurement probes) is the core of identifiability problem, which is associated with two follow-up problems: *monitor placement* and *measurement paths construction* for identifying the objects.

With different assumptions on link metric, existing work could be categorized into 2 classes: algebraic-based approach or statistical-based approach. Algebraic-based approaches assume link metrics to be "constant", implying that either the metric changes slowly relative to the measurement process or it represents statistical characteristics (e.g., mean) that stay constant over time. Algebraic-based approaches leverage algebraic knowledge to compute link metrics from a linear system formed by measurement paths and their metrics [2, 9, 22, 23, 42, 50]. Statistical-based approaches model link metrics as random variables from a family of distribution. For example, [45] assumes such distribution is known and focuses on inferring the parameter of the distribution; [30] uses multicast traffic to infer the delay distribution; [8] applies Fourier transform of the observable distributions to retrieve the unobservable distributions. In this thesis, all the problems investigated fall into the algebraic-based network performance tomography.

[1] Identifiable means the value on a link/path can be uniquely determined.

Complete Network Identifiability

Extracting as much information as possible with available measurement paths is the focus of early work [6,11], however, plenty of later work shows that it is frequently challenging to uniquely identify all the link characteristics [2, 8, 9, 23, 28, 38, 49].

For the networks assuming directed link (links have asymmetrical metrics in different directions), [45] proves that the whole network cannot be identifiable unless every link is directly connected to two monitors. For the networks assuming undirected link (links have symmetric metrics in different directions), Chen *et al.* [9] show the difficulty of uniquely identifying link values and propose to use QR decomposition to solve the challenges caused by linearly dependent paths. Ma *et al.* [28] show that it is generally impossible to identify all the links with two monitors.

Before the growing interest in exploring the *topological conditions* for ensuring network identifiability, there is a trend utilizing Round Trip Time (RTT) to infer link metrics based on the symmetric properties of undirected network [3, 15, 25, 27]. RTT implies the assumption that Internet Control Message Protocol (ICMP) must be adopted by every network node (i.e., router). Nevertheless, this hidden assumption cannot be justified in practice since for security reasons ICMP may be disabled in many routers. Due to this reason, more and more work focuses on establishing the relation between network identifiability and network topology/monitor placement, with different assumptions on measurement paths. Under the assumption that cyclic measurement paths are allowed, [21] is the first work that gives necessary and sufficient topological requirement for all the links to be identifiable. [23] proves that the maximal number of independent equations obtained by measuring cyclic-path delays in an N-node connected network falls behind the number of variables (links) by $(N-1)$. On the account of forming cycles along routing is generally prohibitive in real networks, cycle-free measurement paths are usually assumed [28, 32, 34]. With this assumption, Ma *et al.* [28] derive the necessary and sufficient conditions for a link to be identifiable. They further develop algorithms [34] for computing the link values.

Partial Network Identifiability

Since identifying all network links is not always possible, partial network identifiability is a more practical goal [14, 20, 27, 32, 33, 35, 49]. [20] studies the identifiability problem of a set of preferential/interested links based on cycle-free measurement paths. It trims the graph and then uses an existing method [31] to assign monitors in the trimmed components to identify all links in the interested set. On the top of [20], [14] develops a novel algorithm, Optimal Monitor Assignment (OMA), to assign monitors based on graph partition. In [49], a link sequence of minimal length is set as the inferring target from end-to-end measurements in both directed and undirected networks. [35] proposes a novel approach to approximating the relative value of some link weights. Given a limited number of monitors, [33] develops a polynomial-time greedy algorithm to maximize the partial link identification.

Monitor Placement and Measurement Paths Construction

Once link identification, the natural follow-up question is to place (minimum) monitors and construct paths to perform effective measurement. Ma *et al.* [31] propose an efficient algorithm, Minimum Monitor Placement(MMP), to assign the minimum number of monitors. They also present a spanning tree-based path construction method [34] to construct linearly independent cycle-free paths (i.e., the equations built with these paths are linearly independent). Zheng et al. [50] propose an algorithm to select the minimum number of probing paths that can uniquely determine all identifiable links and cover all unidentifiable links. Along with identifying interested/preferential links, [14] provides a monitor placement strategy, Optimal Monitor Assignment (OMA), that uses the minimum number of monitors to identify all interested links. Ma et al. [33] also propose a near optimal algorithm to achieve the maximum identifiability under a scenario where the number of monitors is given. Furthermore, [42] proposes robust network tomography approach to tolerate link failures by selecting measurement paths with the maximum rank (i.e., the matrix built with the selected paths has the highest rank).

2.2 Problem Formulation

Different assumptions on routing policy result in different hardness for problems in network tomography. In this regard, uncontrollable routing means the routing decision is made purely by the routing protocol based on route-selection criterion, e.g., the shortest-path routing. Controllable routing means that we can manually determine the route using the source-routing mechanism, i.e., the complete path is set in advance in the IP header. On the one hand, [4, 27] show that the problem of minimum monitor placement under uncontrollable routing is NP-hard, and the NP-hardness persists even some network elements are able to control their local routing policy [25]. On the other hand, [31, 32, 34] establish several positive results in linear time under controllable cycle-free routing. Controllable cycle-free routing means that the probing packets sent from monitors follow some specified path (without cycle) using source routing. Source routing is broadly support in single-ISP networks and Software-Define Network (SDN) based WANs. SDN is a new emerging network architecture that is aimed at overcoming the limitation of traditional networking. It has a centralized controller that has knowledge of the topology of the network and is responsible for configuring the forwarding tables on routers. Using source routing, SDN controller can easily dictate paths of measurement packets in the route-setting stage, and the cycle-free requirement precludes endless cycles in the data forwarding stage.

In this thesis, we set all the investigated problems in the context of *controllable*, *cycle-free* measurement paths, i.e., monitor nodes can send probing packets along an arbitrary path to another monitor as long as the path does not form a cycle. Formally, we have the following assumptions:

- Network topology is known and modeled as an undirected, connected graph $\mathcal{G} =< V, L >$, where V and L are the set of nodes and links, respectively.
- All links metrics are additive and constant. The "constant" assumption implies that either the metric changes slowly relative to the measurement process, or it represents a statistical

characteristic (e.g., mean) of the link that remains constant within the time window under our consideration.

- Denote the link incident to node u and v by $l_{u,v}$. Then values on link $l_{u,v}$ and $l_{v,u}$ are the same (i.e., symmetric).
- Each link in $\mathcal{G}$ has two distinct end-points, i.e., no self-loop, and there is at most one link connecting a pair of nodes.
- Certain nodes in V are monitors that can initiate/collect measurements.
- A path is called a measurement path (MP) if its two end-points are monitors. All measurement paths are simple paths, i.e., every node on the path is distinct.

2.2.1 An Illustrative Example

A set of MPs form a linear system. We use the example in Fig. 2.1 to illustrate the concept. The network has 2 monitors in red color. First, the end-to-end additive metric (e.g., delay) along all MPs between the two monitors form the following linear system, where $x_{i,j}$ denotes the additive metric on link $l_{i,j}$ and w_p denotes the end-to-end delay on MP p.

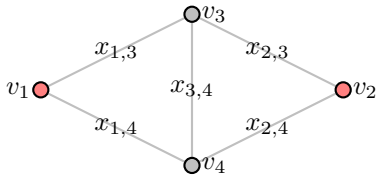


Figure 2.1: An example network.

$$\begin{cases} x_{1,3} + x_{2,3} = w_1 \\ x_{1,4} + x_{2,4} = w_2 \\ x_{1,3} + x_{3,4} + x_{2,4} = w_3 \\ x_{1,4} + x_{3,4} + x_{2,3} = w_4 \end{cases} \tag{2.1}$$

The above linear system could be written into

$$\mathbf{R}\mathbf{x} = \mathbf{w},$$

where

$$\mathbf{R} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix} \tag{2.2}$$

$$\mathbf{x} = \begin{pmatrix} x_{1,3} & x_{2,3} & x_{3,4} & x_{1,4} & x_{2,4} \end{pmatrix}^\intercal \tag{2.3}$$

$$\mathbf{w} = \begin{pmatrix} w_1 & w_2 & w_3 & w_4 \end{pmatrix}^{\intercal} \tag{2.4}$$

$\mathbf{R} = (R_{ij})$ is a $m \times n$ *measurement matrix*, with each entry $R_{ij} \in \{0, 1\}$ denoting whether link l_j is present on path $\mathcal{P}_i$. In general, network tomography needs to deal with the problem of inverting this linear system to obtain $\mathbf{x}$ given $\mathbf{R}$ and $\mathbf{w}$. More often than not, the rank of $\mathbf{R}$ is lower than the number of unknown variables in $\mathbf{x}$.

Chapter 3

Path Identifiability and Optimal Monitor Placement in Concentrated Network Tomography

3.1 Overview

In the "network-as-a-service" paradigm, network operators have a strong need to know the metrics of critical paths running services to their users/tenants. However, it is usually prohibitive to directly measure the metrics of all such paths due to the measuring overhead. A practical solution is to use network tomography to infer the metrics of such paths based on observations from a small number of monitoring nodes. This problem could be termed as path identifiability problem, a new problem that largely differs from existing link identifiability problems. In this chapter, we show that the new problem is harder than link identifiability problems, in the sense that fewer monitors are required for identifying the metrics of given paths than for identifying the metrics of links along the paths. To solve the problem, we develop sufficient and necessary conditions for the identifiability of a given set of interested paths, and design an efficient algorithm that deploys the minimum number of monitors. Experiments show a saving of up to 40% fewer monitors that guarantee the identifiability of a given set of paths.

3.2 Problem Description and High-level Ideas

Consider a network $\mathcal{G} =< V, L >$, where V and L are the sets of nodes and links in $\mathcal{G}$, respectively. Let l_{v_i,v_j} denote the link between nodes v_i and v_j, and $x_{i,j}$ denote its additive metric, e.g., delay, where we assume that $x_{i,j}$ and $x_{j,i}$ have the same value. We assume $\mathcal{G}$ is a simple graph, that is, the end points of each link are different and each link only occurs once in L. We assume that any node in

the network $\mathcal{G}$ can be selected to be a *monitor*, which can send and receive end-to-end measurements and record related measurement information regarding a performance metric. All the measurement paths are cycle-free. All these assumptions are the same as in [14, 31].

Definition 1. ***(Identifiability)****: A path p is called* identifiable *if the sum of corresponding additive metric of each link in p can be obtained. Similarly, a link l is called identifiable if the metric of l can be obtained.*

The problem to be solved in this chapter is defined as follows:

Problem 1. ***(Optimal Monitor Placement (OMP))****: Given a network $\mathcal{G}$ and a set of paths $\mathbf{P}$ in the network, place the minimum number of monitors so that with the measurements among those monitors, all paths in $\mathbf{P}$ are identifiable.*

In order to solve the **OMP** problem, we first solve the following sub-problem:

Problem 2. ***(Path Identification Problem (PIP))****: Given a set of monitors and a path p, determine whether or not p is identifiable with the given monitors.*

The solution of PIP (Section 3.3) is the basic building block for solving OMP (Section 3.4). In the solution of PIP, we decompose the graph into special types of subgraphs (mainly, bi-connected component (BC) and tri-connected components (TC)), and find the conditions to identify paths within and across the subgraphs. This result allows us to develop an optimal algorithm that iteratively selects a small number of initial monitors (i.e., at least these monitors are required to identify paths in $\mathbf{P}$), and adds extra monitors as needed until all paths in $\mathbf{P}$ are identified.

Table 3.1: **Notations**

Symbol	Meaning
V	set of nodes in graph $\mathcal{G}$
L	set of links in graph $\mathcal{G}$
$V(\mathcal{G}')$, $L(\mathcal{G}')$	set of nodes/links in subgraph $\mathcal{G}'$
$\mathbf{P}$	set of Interested Paths
l_{v_i,v_j}	the link between v_i and v_j
$E(p)$	set of end nodes of a path p
$V(p)$, $L(p)$	set of nodes/links in a path p
$L(v)$	set of links incident to node v
$\mathcal{H}(\mathcal{G})$	interior graph of $\mathcal{G}$
$d(v)$	degree of v

The key notations are summarized in Table 3.1. Here we introduce some basic concepts in graph theory, which are needed to make our discussion clear. In a graph $\mathcal{G}$,

- *cut-point*: a node whose removal disconnects $\mathcal{G}$.
- *2-vertex-cut*: a pair of nodes whose removal disconnects $\mathcal{G}$, but $\mathcal{G}$ is connected if only one of them is removed.
- *cut node*: a node that is a cut-point or one of a 2-vertex-cut in $\mathcal{G}$.

- *k-vertex-connected*: a graph which has more than k nodes and remains connected unless at least k nodes are removed.
- *BC*: a maximal subgraph of $\mathcal{G}$ that is either (i) a single link, or (ii) 2-vertex-connected.
- *TC*: a maximal subgraph of $\mathcal{G}$ that is either (i) a circle, or (ii) 3-vertex-connected.
- *2-bridge-cut*: a pair of links whose removal increases its number of connected components. And $\mathcal{G}$ is connected if only one of them is removed.
- *interior graph*: a subgraph of $\mathcal{G}$ containing only 2 monitors, which is obtained by removing the 2 monitors and their incident links.
- *exterior links*: the links incident to only 1 monitor in $\mathcal{G}$ containing only 2 monitors.
- *vantage w.r.t. a TC* $\mathcal{T}$: a node that is either (i) a monitor in $\mathcal{T}$, or (ii) a cut node that separates $\mathcal{T}$ from at least one monitor.
- *Edge-BC*: a BC including only one cut-point.
- *Edge-TC*: a TC including only one 2-vertex-cut.

3.3 Path Identification with Given Monitors

In this section, we investigate **Path Identification Problem (PIP)** (determine the path identifiability with given monitors).

At the high-level, PIP is solved in four steps:

1. First, an extended graph $\mathcal{G}_{new}$ with 2 virtual monitors is constructed based on $\mathcal{G}$, converting any k-monitor ($k >= 2$) problem in $\mathcal{G}$ into a 2-monitor problem in $\mathcal{G}_{new}$. For any link/path in $\mathcal{G}$, it bears the same identifiability in $\mathcal{G}_{new}$.
2. Second, $\mathcal{G}_{new}$ is partitioned into special types of components to facilitate analysis on link/path identifiability.
3. Third, based on graph partition, the identifiability of each special link type in each component is analysed.
4. Finally, we classify a path into different cases, and find the conditions for path identifiability in each case.

Step 1: Graph Extension

Given a network $\mathcal{G}$ containing k monitors ($k \geq 2$), we construct an extended graph $\mathcal{G}_{new}$ (e.g., Fig. 3.1) as follows: (1) Add two virtual monitors m_1' and m_2' to $\mathcal{G}$. (2) Add a virtual link between a virtual monitor and an existing monitor (also called a real monitor) in $\mathcal{G}$. (3) Add a virtual link between the two virtual monitors. The first two steps above are the same as in [31]. In our extended

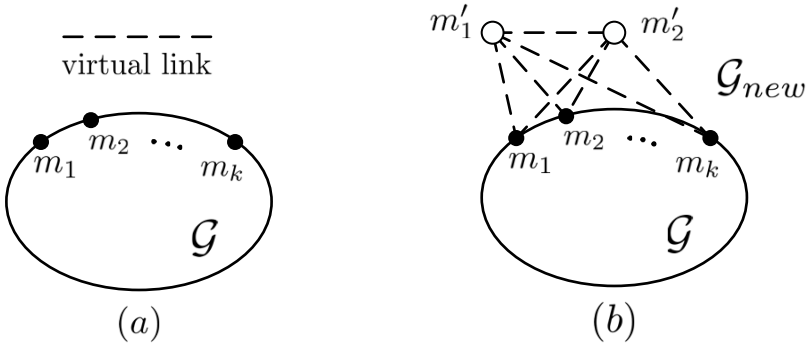


Figure 3.1: (a) $\mathcal{G}$ with k monitors ($k \geq 2$), (b) $\mathcal{G}_{new}$ of $\mathcal{G}$ with 2 virtual monitors.

graph, however, we add a virtual link between the two virtual monitors which enables m'_1 and m'_2 to be included in a TC of $\mathcal{G}_{new}$ when $k = 2$. The details will be explained in Step 2.

Since $\mathcal{G}$ is the *interior graph* of $\mathcal{G}_{new}$, any measurement between the real monitors can be obtained from measurements between m'_1 and m'_2, and the measurements between m'_1 and m'_2 in $\mathcal{G}_{new}$ do not provide extra information for identifying links in $\mathcal{G}$ compared with measurements generated by the real monitors [31]. Therefore, the identifiability of any path/link in $\mathcal{G}$ is the same as in $\mathcal{G}_{new}$.

This step converts the problem for determining a path's identifiability in $\mathcal{G}$ with k monitors to the problem of determining a path's identifiability in $\mathcal{G}_{new}$ with only two (virtual) monitors. As such, in the rest of this section, we focus on a path's identifiability with two (virtual) monitors. In addition, any path in $\mathcal{G}$ is taken as a path including no virtual monitors in $\mathcal{G}_{new}$.

Step 2: Graph Decomposition

To study the identifiability of paths/links in $\mathcal{G}_{new}$, we first partition $\mathcal{G}_{new}$ into bi-connected components (BC) by the algorithm in [41]. m'_1 and m'_2 must be included in the same BC (denoted by $\mathcal{B}_0$) of $\mathcal{G}_{new}$, due to the virtual link between m'_1 and m'_2. Any link $l \notin L(\mathcal{B}_0)$ is unidentifiable, because l cannot be included in any measurement path between m'_1 and m'_2. Therefore, we mainly focus on $\mathcal{B}_0$, and then partition it into TCs.

Based on the connectivity between the virtual monitors and the real monitors, the following conclusions can be drawn:

- m'_1 and m'_2 are included in the same TC, denoted by $\mathcal{T}_0$;
- each TC $\mathcal{T}$ of $\mathcal{B}_0$ includes only two vantages. The vantages of $\mathcal{T}_0$ are m'_1 and m'_2, and each of the other TCs includes only one 2-vertex-cut that separates itself from m'_1 and m'_2.
- All the TCs of $\mathcal{B}_0$ can be arranged in a *tree* structure, if we treat $\mathcal{T}_0$ as the *root*, the other TCs as *nodes*, and the 2-vertex-cut between the TCs as the *edge*. An example is shown in Fig. 6.1.

Therefore, we can group TCs according to their distance to $\mathcal{T}_0$ along the tree.

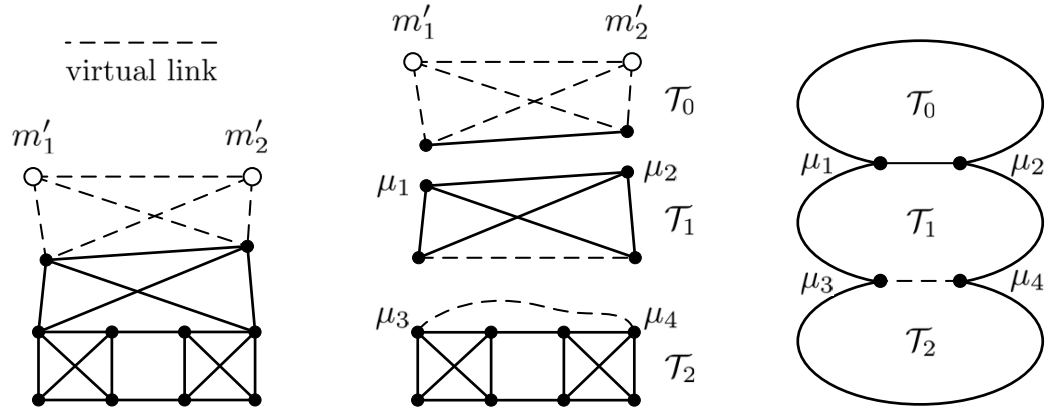


Figure 3.2: An example of graph decomposition. Note that μ_1 and μ_2 are the vantages w.r.t. $\mathcal{T}_1$, and μ_3 and μ_4 are the vantages w.r.t. $\mathcal{T}_2$.

Step 3: Identifying Links in a TC

After Step 2, we study the identifiability of links in a TC with two vantages, excluding the direct link between the two vantages (w.r.t. this TC). Given a TC $\mathcal{T}$ with two vantages μ_1 and μ_2, let $\widetilde{\mathcal{T}}$ denote the subgraph of $\mathcal{T}$ obtained by removing the direct link between μ_1 and μ_2. We have the following theorem.

Theorem 1. *For a 3-vertex-connected TC $\mathcal{T}$ with two vantages, the following conclusions can be drawn about the identifiability of links in $\widetilde{\mathcal{T}}$:*

1. *All of the exterior links are unidentifiable;*
2. *If $\mathcal{T}$ is 3-vertex-connected and $\widetilde{\mathcal{T}}$ is not 3-vertex-connected, then all of the interior links are identifiable except the links that belong to the 2-bridge-cuts in $\widetilde{\mathcal{T}}$; otherwise, all of the interior links are identifiable.*

An Illustrative Example of Link Identifiability

As an example, Fig. 3.3 shows a TC, of which each link's identifiability could be determined by Theorem 1. Note that the identifiability of the direct link between the two vantages can be determined in the parent TC which they belong to. In the TC tree shown in Fig. 3.2, μ_3 and μ_4 are the vantages w.r.t. $\mathcal{T}_2$, but they are not the vantages w.r.t. $\mathcal{T}_1$. Their identifiability can be determined in $\mathcal{T}_1$, following the above procedure.

Remark 1. *While the identifiability of links in TC has been studied in [20] (i.e.,Theorem III.3 in [20]), the theorem in [20] did not consider a special case where there is a 2-bridge-cut in $\widetilde{\mathcal{T}}$. Theorem 1 in this paper points out that links in 2-bridge-cut are not identifiable, which may affect path identifiability as to be discussed shortly.*

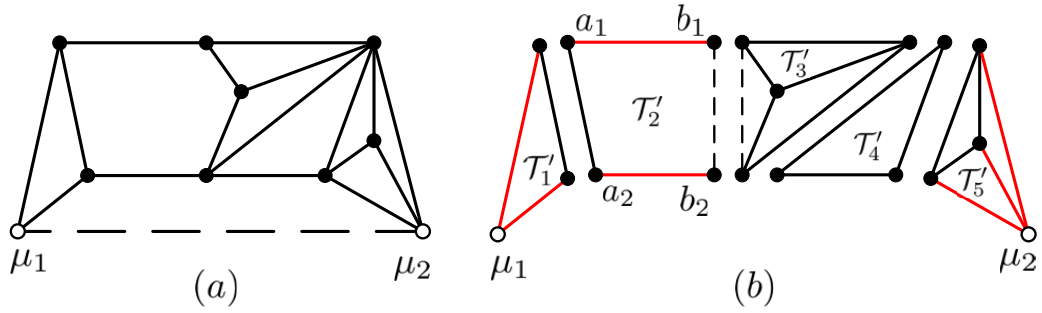


Figure 3.3: (a) $\mathcal{T}$ with 2 vantages μ_1 and μ_2; (b) $\mathcal{T}_1', \ldots, \mathcal{T}_5'$ are TCs within $\widetilde{\mathcal{T}}$, $\mathcal{T}_2'$ includes one 2-bridge-cut. In $\widetilde{\mathcal{T}}$, only the links in red are unidentifiable.

Step 4: Determining a Path's Identifiability

We stress again that to identify a path, it is not necessary to identify every link along the path. Nevertheless, we need to identify some links, whenever necessary, using the results from previous three steps.

Based on the decomposition of $\mathcal{G}_{new}$ and previous results, a path p in $\mathcal{G}$ can be categorized into one of the following three cases:

- **category 1**: $V(p) \not\subseteq V(\mathcal{B}_0)$;
- **category 2**: $V(p) \subseteq V(\mathcal{B}_0)$ and $E(p)$ are not in the same TC;
- **category 3**: $V(p) \subseteq V(\mathcal{B}_0)$ and $E(p)$ are in the same TC.

Theorem 2. *If a path p belongs to **category 1** or **category 2**, then p is unidentifiable.*

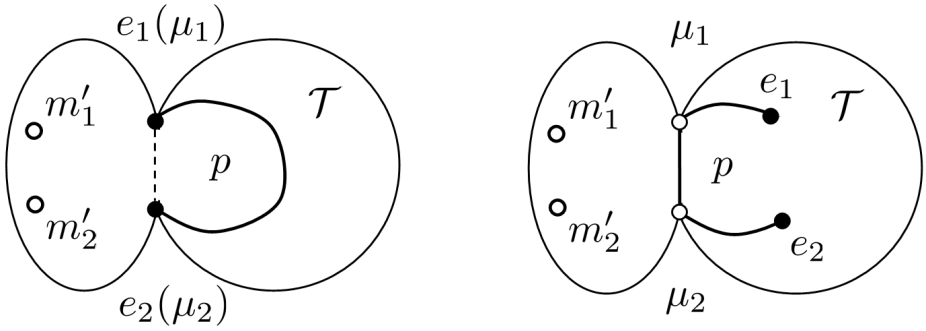
Proof. Refer to Section 3.6.2. ■

If a path p belongs to **category 3**, p can be classified into two cases: $V(p)$ is in one TC or $V(p)$ is not in one TC. In the rest of this section, we present path identifiability for such two cases.

Case 1: $V(p)$ is in one TC

Since any TC $\mathcal{T}$ in $\mathcal{G}_{new}$ includes 2 vantages μ_1 and μ_2, a path p satisfying that $V(p) \subseteq V(\mathcal{T})$ can be categorized into one of the following three cases:

- *D-shape*: $\{\mu_1, \mu_2\} \subseteq V(p)$ and $E(p) = \{\mu_1, \mu_2\}$
- *U-shape*: $\{\mu_1, \mu_2\} \subseteq V(p)$ and $E(p) \neq \{\mu_1, \mu_2\}$
- *single-TC*: $\{\mu_1, \mu_2\} \not\subseteq V(p)$



(a) D-shape (b) U-shape

Figure 3.4: (a) D-shape. (b) U-shape. Note that e_1 and e_2 are the two end points of path p.

As shown in Fig. 3.4, if p belongs to *D-shape* or *U-shape*, p must include the unidentifiable links, which are the exterior links of $\mathcal{T}$. The identifiability of p, however, can be determined with Lemma 1, which does not rely on the identifiability of each link in p.

Lemma 1. *Assume that the two vantages of a TC $\mathcal{T}$ are μ_1 and μ_2, and l_{μ_1,μ_2} is the direct link*[1] *between vantages. The following conclusions can be drawn:*

1. *a D-shape path p is identifiable, if and only if l_{μ_1,μ_2} is identifiable;*
2. *a U-shape path p is identifiable, if and only if both l_{μ_1,μ_2} and p_e^s are identifiable, where p_e^s is the shortest path between $E(p)$ satisfying that $(V(p_e^s) - E(p_e^s))$ does not include any vantage, i.e., the nodes along p_e^s (excluding the end nodes) do not include any vantage.*

Proof. Refer to section 3.6.2. ■

If p belongs to *single-TC*, then the identifiability of p can be determined with Theorem 3.

Theorem 3. *Given a TC $\mathcal{T}$ with two vantages, if a path p belongs to* single-TC*,*

1. *if $\mathcal{T}$ is a circle, p is unidentifiable;*
2. *if $\mathcal{T}$ is 3-vertex-connected, p is identifiable if and only if every link in $L(p)$ is identifiable.*

Proof. Refer to Section 3.6.2. ■

Case 2: $V(p)$ not in a TC

In this case, we need to further classify p into different sub-cases. For this, we introduce two new notations, $\mathcal{T}_{V_{min}}$ and $\mathcal{T}_{E_{min}}$.

Definition 2. *Given a path p belonging to* **category 3**, *$\mathcal{T}_{V_{min}}$ is a TC $\mathcal{T}$ that satisfies the following two conditions:*

1. *$|V(\mathcal{T}) \cap V(p)| \geq 2$;*

[1]If there is no direct link between μ_1 and μ_2 in $\mathcal{G}$, we can add a virtual direct link between them without any impact on the identifiability of p.

2. the parent TC $\mathcal{T}'$ of $\mathcal{T}$ satisfies that $|V(\mathcal{T}') \cap V(p)| < 2$.

Definition 3. *Given a path p belonging to* **category 3**, *$\mathcal{T}_{E_{min}}$ is a TC $\mathcal{T}$ that satisfies the following two conditions:*

1. $|V(\mathcal{T}) \cap E(p)| = 2$;

2. the parent TC $\mathcal{T}'$ of $\mathcal{T}$ satisfies that $|V(\mathcal{T}') \cap E(p)| < 2$.

Intuitively, $\mathcal{T}_{V_{min}}$ is the TC that includes at least two nodes of p and is closest to the root (including the root) on the tree structure presented in *Step 2*. $\mathcal{T}_{E_{min}}$ is the TC that includes the two end points of p and is closest to the root (including the root) on the tree structure. Here, Let V_{min} (E_{min}) denote the distance between $\mathcal{T}_{V_{min}}$ ($\mathcal{T}_{E_{min}}$) and $\mathcal{T}_0$ along the *tree*. We have $V_{min} \leq E_{min}$, due to $E(p) \subseteq V(p)$.

Therefore, a path p in **Category 3-Case 2** can be classified into one of the following two types:

- p is of **hill** if $V_{min} = E_{min}$ (e.g., Fig. 3.5 (a));
- p is of **pipe** if $V_{min} < E_{min}$ (e.g., Fig. 3.5 (c)).

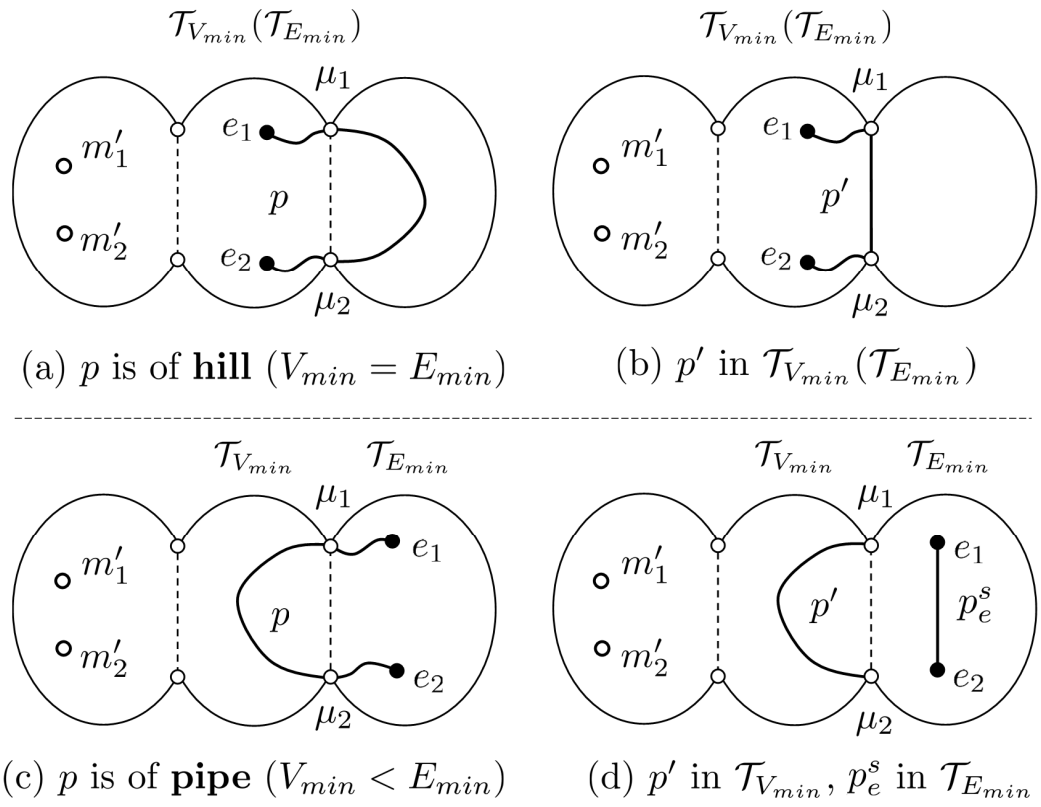


Figure 3.5: Two types of p in *category 3-case 2*: (a) hill, (b) transform of hill, (c) pipe, (d) transform of pipe.

We need to define two path transforms to handle the hill path and the pipe path, respectively.

- **Transform 1:** For a hill path p, we replace the sub-path of p in the child TC of $\mathcal{T}_{V_{min}}$ with link l_{μ_1,μ_2}, where μ_1, μ_2 are the vantages w.r.t. the child TC of $\mathcal{T}_{V_{min}}$. As an example, the hill path p in Fig. 3.5 (a) after transform 1 becomes p' in Fig. 3.5 (b).
- **Transform 2:** For a pipe path p, we replace p with two paths, one is the sub-path of p included in $\mathcal{T}_{V_{min}}$, the other is the shortest path between e_1 and e_2, denoted by p_e^s, satisfying the condition that $(V(p_e^s) - E(p_e^s))$ does not include any vantage. As an example, the pipe path p in Fig. 3.5 (c) after transform 2 becomes p' and p_e^s in Fig. 3.5 (d).

Theorem 4. *For a path p in* **Category 3-Case 2***:*

- *if p is a* **hill** *path: p* is identifiable if and only if p' is identifiable.
- *if p is a* **pipe** *path: p* is identifiable if and only if both p' and p_e^s are identifiable.

Since p' and p_e^s both belong to Case 1- single-TC*, their identifiability can be determined with Theorem 3.*

Proof. Refer to Section 3.6.2. ■

To summarize, the flow chart of identifying a path p is shown in Fig 3.6.

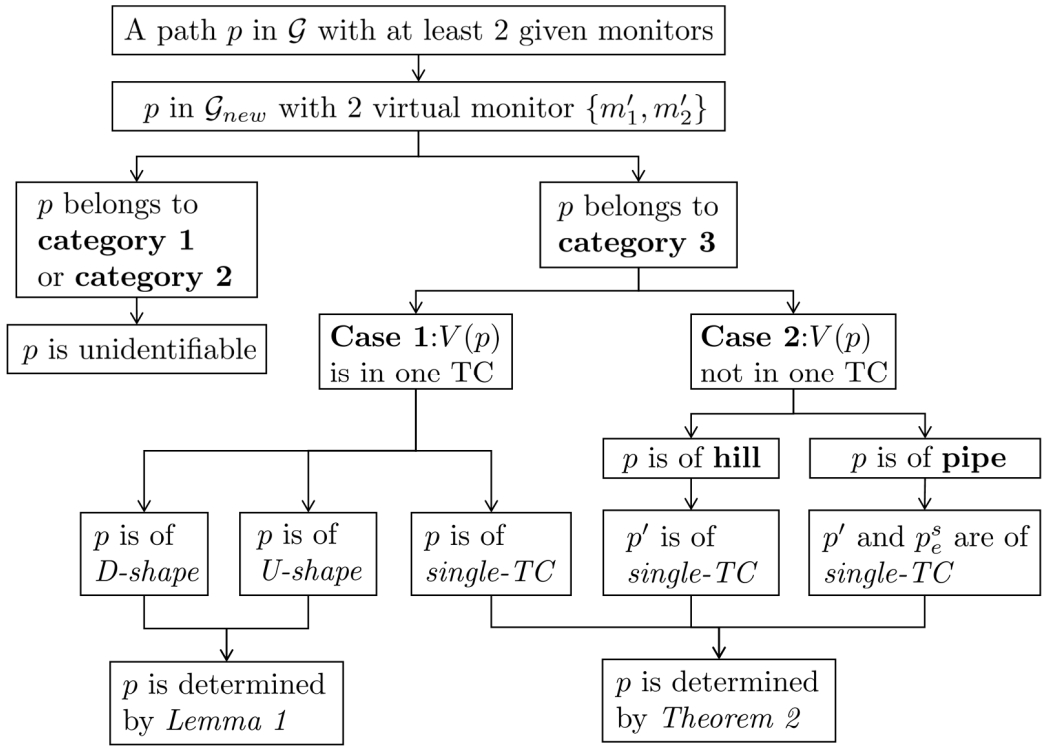


Figure 3.6: The identifiability of path p.

3.4 Monitor Placement for Identifying Interested Paths

In this section, we solve the **OMP** problem. Based on the results in the previous section, we first present a sufficient and necessary condition for identifying a path in the extended graph for a graph containing at least two monitors. Then we develop an optimal algorithm to minimize the number of extra monitors needed in order to identify a set of interested paths in a graph when its extended graph is 2-vertex-connected. Finally, given a graph $\mathcal{G}$ and $\mathbf{P}$, we develop an optimal algorithm to identify all the paths in $\mathbf{P}$ with the minimum number of monitors by iteratively selecting initial monitors, converting $\mathcal{G}_{new}$ into a single 2-vertex-connected graph, and calling for the optimal monitor placement algorithm in a single 2-vertex-connected graph.

3.4.1 Necessary and Sufficient Condition for Identifying a Path in a Graph with at Least 2 Monitors

According to the previous section, all paths could be classified into three categories: **category 1**, **category 2** or **category 3**. Paths belonging to **category 1** or **category 2** are not identifiable unless extra monitors are added. If a path belongs to **category 3-case 1**, its identifiability can be determined with Lemma 1 (*D-shape, U-shape*) or Theorem 3 (*single-TC*). If a path belongs to **category 3-case 2**, either a hill or a pipe, its identifiability can be converted to the identifiability of one path or two paths, whose type is *single-TC*, as shown in Fig. 3.5. The identifiability of paths in *single-TC* depends on the conditions stated in Theorem 3.

Assume that we add an extra monitor m in $\mathcal{G}$. Accordingly, we need to update the extended graph $\mathcal{G}_{new}$ by adding two virtual links between m and the virtual monitors. As before, we use $\mathcal{T}_0$ to denote the TC that includes the two virtual monitors.

A sufficient and necessary condition for identifying a path in a graph with at least 2 monitors is given in Theorem 5:

Theorem 5. *Given a graph $\mathcal{G}$ with at least two monitors, if p is not identifiable and p satisfies that (i) its end nodes are in different BCs (a case listed in* **category 1***), or (ii) it belongs to* **category 2***, or (iii) it belongs to* single-TC*, then p can be identified if and only if extra monitors are assigned so that both the end nodes of p are included in $\mathcal{T}_0$.*

3.4.2 Monitor Placement for Identifying a Set of Paths in a 2-vertex-connected Graph with at Least Two Initial Monitors

Given a 2-vertex-connected graph $\mathcal{G}$ with at least two monitors, we develop an optimal algorithm to identify all paths in $\mathbf{P}$ with the minimum number of extra monitors. We first construct the extended graph $\mathcal{G}_{new}$ of $\mathcal{G}$, which includes only one BC $\mathcal{B}_0$ since $\mathcal{G}$ is 2-vertex-connected. And then we partition $\mathcal{G}_{new}$ into TCs.

For an Edge-TC $\mathcal{T} \in \mathcal{G}_{new}$, following the tree structure defined in Step 2 of the previous section, there are a set of 2-vertex-cuts and a set of TCs between $\mathcal{T}$ and $\mathcal{T}_0$. If $\mathcal{T}$ has been assigned an extra monitor, then the nodes in 2-vertex-cuts and 3-vertex-connected TC between $\mathcal{T}$ and $\mathcal{T}_0$ will be included in $\mathcal{T}_0$. Therefore, for node $v \notin \mathcal{T}_0$, depending on the type of $\mathcal{T}$, Lemma 2 and Lemma 3 correspondingly provide the sufficient and necessary condition for v being included in $\mathcal{T}_0$.

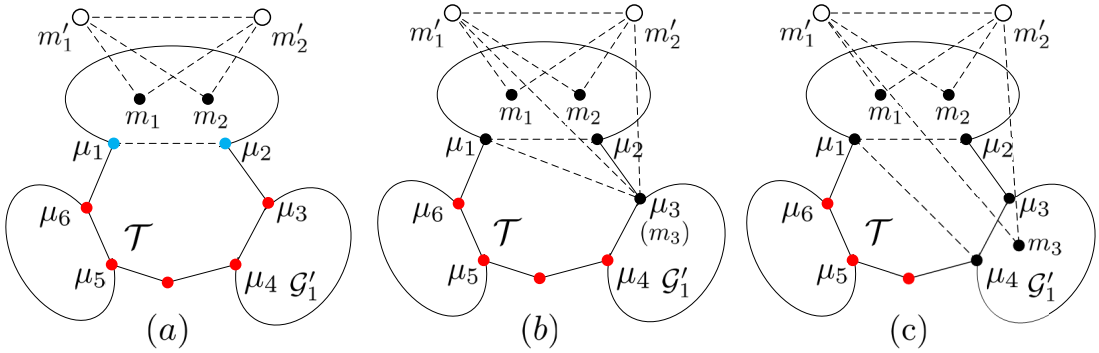


Figure 3.7: Monitor assignment for circular TC $\mathcal{T}$ ($\mathcal{H}(\mathcal{T})$ denoted in red).

Lemma 2. *For a circular TC $\mathcal{T}$ in $\mathcal{G}_{new}$, a node $v \in V(\mathcal{H}(\mathcal{T}))$ is included in $\mathcal{T}_0$ if and only if an extra monitor m is assigned so that :*

1. *$m = v$; or*
2. *$m \notin V(\mathcal{T})$, but assigned in a subgraph of $\mathcal{G}_{new}$, which does not include $\mathcal{T}$ and is obtained from $\mathcal{G}_{new}$ by removing v and its neighboring node in $\mathcal{T}$.*

Proof. Refer to Section 3.6.2. ■

We use Fig. 3.7 to illustrate Lemma 2. There is a node $\mu_3 \notin \mathcal{T}_0$ in Fig. 3.7 (a). If an extra monitor m_3 is assigned at μ_3 as in Fig. 3.7 (b) or in the subgraph as in Fig. 3.7 (c), then $\mu_3 \in \mathcal{T}_0$. However, for a circular TC $\mathcal{T}$, if two nodes need to be included in $\mathcal{T}_0$, the number of extra monitors may be different. For example, in Fig. 3.7, if μ_3 and μ_5 need to be included in $\mathcal{T}_0$, then two extra monitors need to be placed. If μ_3 and μ_4 need to be included in $\mathcal{T}_0$, then only 1 monitor, m_3, needs to be placed as in Fig. 3.7 (c). Algorithm 1 is developed to handle monitor placement in circular TCs.

Algorithm 1 Monitor Placement for circular TCs

Input: A Circle $\mathcal{T}$, a path set $\mathbf{P}$
Output: $M_{\mathcal{T}}$

$\{\mu_1, \mu_2\}$ = a 2-vertex-cut of $\mathcal{T}$, and $n = |V(\mathcal{H}(\mathcal{T}))|$;
let $v'_1, ..., v'_n$ be the vertices in $V(\mathcal{H}(\mathcal{T}))$ from μ_1 to μ_2;
$i = 1$
while $i \leq n$ **do**
  if $v'_i \in E(\mathbf{P})$ **and** $d(v'_i) = 2$ **then**
    if $v'_{i+1} \in E(\mathbf{P})$ **and** $d(v'_{i+1}) = 2$ **and** $l_{v'_i, v'_{i+1}}$ has an assistant node w **then**
      $M_{\mathcal{T}} = M_{\mathcal{T}} \cup \{w\}$; $i = i + 2$;
    else
      $M_{\mathcal{T}} = M_{\mathcal{T}} \cup \{v'_i\}$; $i = i + 1$;
  else
    $i = i + 1$;
return $M_{\mathcal{T}}$

Lemma 3. *For a 3-vertex-connected TC $\mathcal{T}$ in $\mathcal{G}_{new}$, $V(\mathcal{H}(\mathcal{T})) \subseteq V(\mathcal{T}_0)$ if and only if an extra monitor m is added subject to that:*

1. *m is not a vantage of $\mathcal{T}$, and*

2. *m is in a subgraph of $\mathcal{G}_{new}$, which includes $\mathcal{T}$ and is obtained from $\mathcal{G}_{new}$ by removing the vantages of $\mathcal{T}$.*

Proof. Refer to Section 3.6.2. ■

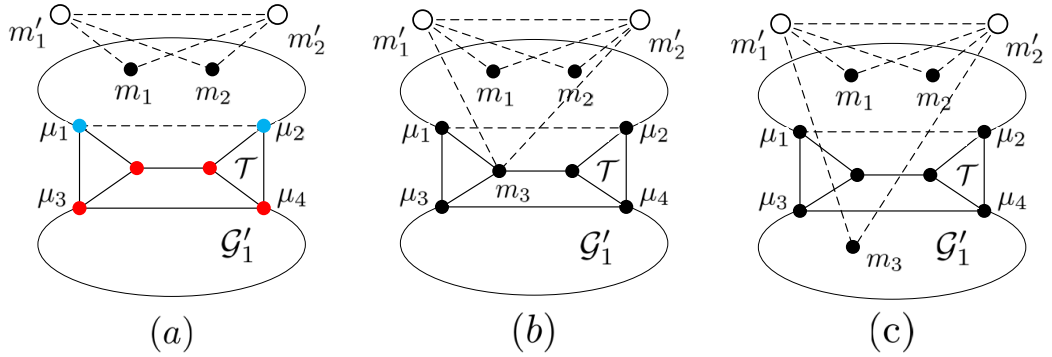


Figure 3.8: Monitor assignment for non-circular TC ($\mathcal{H}(\mathcal{T})$ denoted in red).

We use Fig. 3.8 to illustrate Lemma 3. The TC in consideration is marked in red in Fig. 3.8 (a). If an extra monitor m_3 is assigned at a non-vantage point as in Fig. 3.8 (b) or in the subgraph as in Fig. 3.8 (c), then all the nodes in the interior graph will be included in $\mathcal{T}_0$, i.e., $V(\mathcal{H}(\mathcal{T})) \subseteq V(\mathcal{T}_0)$. Therefore, for a 3-vertex-connected TC $\mathcal{T}$, no matter how many nodes in $\mathcal{H}(\mathcal{T})$ need to be included in $\mathcal{T}_0$, one extra monitor is enough.

We now introduce Algorithm 2 to assign monitors so that all paths in $\mathbf{P}$ can be identified in a 2-vertex-connected graph $\mathcal{G}_{new}$. Algorithm 2 works in two steps: first, it assigns monitors for Edge-TCs as needed, and it trims a number of Edge-TCs which need not be assigned monitors, until each remained Edge-TC has been assigned monitors. Second, it assigns monitors for the remained circular TCs. The complexity of Algorithm 2 is $O(|V| + |L| + |\mathbf{P}|)$.

Theorem 6. *For a target $\mathbf{P}$ in a network $\mathcal{G}$ with at least two monitors, if $\mathcal{G}_{new}$ of $\mathcal{G}$ is 2-vertex-connected, then Algorithm 2 can place an optimal number of monitors that identify all paths in $\mathbf{P}$, i.e., (i) all paths in $\mathbf{P}$ can be identified, and (ii) no placement can identify all the paths with a smaller number of extra monitors.*

Proof. Refer to Section 3.6.2. ■

3.4.3 Monitor Placement for Identifying a Set of Paths

Given a graph $\mathcal{G}$ and a path set $\mathbf{P}$, we develop an optimal algorithm to identify all the paths in $\mathbf{P}$ with the minimum number of monitors.

We first partition $\mathcal{G}$ into BCs. If an edge-BC exists and includes no links in $\mathbf{P}$, then the edge-BC can be trimmed, since there is no link that needs to be measured. Hence, we first obtain a trimmed graph $\mathcal{G}^t$ by trimming a set of BCs [20], until each edge-BC of $\mathcal{G}^t$ includes at least one link in $L(\mathbf{P})$. According to the number of BCs in $\mathcal{G}^t$, there are two cases: *only one BC* or *more than one BC*.

Algorithm 2 Monitor Placement with At Least Two Monitors

Input: A graph $\mathcal{G}$, initial monitors M_{in}, target set $\mathbf{P}$

Output: $M(\mathcal{G}, M_{in}, \mathbf{P})$

obtain $\mathcal{G}_{new}$ and partition $\mathcal{G}_{new}$ into BCs [41];
if $\mathcal{G}_{new}$ includes only one BC $\mathcal{B}_0$ **then**
  partition $\mathcal{B}_0$ into SPQR components [13];
  obtain $\mathbf{P}'$ by converting all the hills and pipes in $\mathbf{P}$;
  $C = \{$all circular TCs in $\mathcal{B}_0\}$
  $Q = \{$all Edge-TCs in $\mathcal{B}_0\}$, and $M = M_{in}$;
  while there exists an Edge-TC $\mathcal{T} \in Q$ **do**
    $\{\mu_1, \mu_2\}$ =the 2-vertex-cut of $\mathcal{T}$;
    if $\mathcal{T}$ is circle **and** $E(\mathbf{P}') \cap V(\mathcal{H}(\mathcal{T})) \neq \emptyset$ **then**
      $C = C - \mathcal{T}$, and obtain $M_{\mathcal{T}}$ by Algorithm 1;
      $M = M \cup M_{\mathcal{T}}$;
    else if $E(\mathbf{P}') \cap V(\mathcal{H}(\mathcal{T})) \neq \emptyset$ **then**
      for each path p_j with $E(p_j) \cap V(\mathcal{H}(\mathcal{T})) \neq \emptyset$ **do**
        if $|E(p_j) \cap V(\mathcal{H}(\mathcal{T}))| = 1$ **then**
          $M = M \cup \{v\}, v \in \mathcal{H}(\mathcal{T})$; break;
        else
          if p_j includes unidentifiable links **then**
            $M = M \cup \{v\}, v \in \mathcal{H}(\mathcal{T})$; break;
    if $\mathcal{T}$ includes no monitor **then**
      if $\mathcal{T}$ has only one neighbour TC $\mathcal{T}^n$ **then**
        if $\mathcal{T}^n$ is a circle **then**
          set an assistant node v for $l_{1,2}$ ($v \in \mathcal{H}(\mathcal{T})$);
        set $\{\mu_1, \mu_2\}$ as not 2-vertex-cut;
      delete $\mathcal{T}$ except μ_1, μ_2, set $l_{\mu_1\mu_2}$ as real link;
      if $\mathcal{T}^n$ includes only one 2-vertex-cut **then**
        $Q = Q \cup \mathcal{T}^n$;
  for each circle $\mathcal{T}_i \in C$ **do**
    if there exists a node $v \in E(\mathbf{P})$ and $d(v) = 2$ **then**
      obtain $M_{\mathcal{T}_i}$ by Algorithm 1;
      $M = M \cup M_{\mathcal{T}_i}$;
return $M(\mathcal{G}, M_{in}, \mathbf{P}) = M$

Case 1: $\mathcal{G}^t$ includes *only one BC*. Since at least two monitors are required to generate a measurement path, we can obtain two initial monitors by enumerating all pairs of nodes, and for each pair of initial monitors, Algorithm 2 generates a monitor placement that uses the minimum number of extra monitors. The complexity of the algorithm is $O(|V|^2(|V| + |L| + |\mathbf{P}|))$.

Case 2: $\mathcal{G}^t$ includes *more than one BC*. Each edge-BC $\mathcal{B}$ must be assigned at least one monitor, otherwise each link in $\mathcal{B}$ cannot be measured since it is not included in any measurement path. Therefore, the monitors should be assigned in two steps as follows: First, place monitors for each

edge-BC $\mathcal{B}$; Second, place monitors for $\mathcal{G}$ and $\mathbf{P}$.

In the first step, for each edge-BC $\mathcal{B}$, its cut-point $c_{\mathcal{B}}$ can be considered as an initial monitor, since there must be at least two edge-BCs and then $c_{\mathcal{B}}$ must connect to a monitor not in $\mathcal{B}$. Let $\mathbf{P}_{\mathcal{B}}$ denote the set of interested paths in $\mathcal{B}$. In particular, if $\mathcal{B}$ includes a node, which is the end node v of a multi-BC path p, then the sub-path p' of p between v and $c_{\mathcal{B}}$ should also be included in $\mathbf{P}_{\mathcal{B}}$. We then execute Algorithm 2 to identify paths in $\mathbf{P}_{\mathcal{B}}$. With the initial two monitors, including $c_{\mathcal{B}}$ and one other node selected by enumeration, Algorithm 2 returns the optimal monitor placement (i.e., the one with the minimum number of extra monitors) for $\mathcal{B}$. The complexity for each BC $\mathcal{B}$ is $O(|V(\mathcal{B})|(|V(\mathcal{B})|+|L(\mathcal{B})|+|P(\mathcal{B})|))$, and thus the total complexity of step one is no larger than $O(|V|(|V|+|L|+|\mathbf{P}|))$.

In the second step, when each edge-BC has been assigned monitors, $\mathcal{G}_{new}$ of $\mathcal{G}$ will be 2-vertex-connected since each subgraph of $\mathcal{G}$ includes at least one monitor, which is obtained by removing any cut-point in $\mathcal{G}$. Therefore, a monitor placement for $\mathcal{G}$ can be obtained by Algorithm 2. The complexity of step 2 is $O(|V|+|L|+|\mathbf{P}|)$, then the complexity for $\mathcal{G}^t$ including more than one BC is $O(|V|(|V|+|L|+|\mathbf{P}|))$.

Algorithm 3 Monitor Placement for Interested Paths

Input: A network $\mathcal{G}$ and a target $\mathbf{P}$
Output: $M(\mathcal{G},\mathbf{P})$

obtain trimmed graph $\mathcal{G}^t$ and its BCs by *Algorithm* 1 $(\mathcal{G}, L(\mathbf{P}))$ in [20];
$M(\mathcal{G}^t,\mathbf{P})=\emptyset$
if $\mathcal{G}^t$ includes only one BC **then**
  for each set S_i of two nodes in $\mathcal{G}^t$ **do**
    $S_i = S_i \cup M(\mathcal{G}^t,\mathbf{P})$
    M_i = Algorithm 2 $(\mathcal{G}^t, S_i, \mathbf{P})$;
  $M(\mathcal{G}^t,\mathbf{P}) = \arg\min(|M_i|)$;
else
  for each edge-BC $\mathcal{B}_i$ **do**
    for each node v_j in $\mathcal{B}_i$ except the cut-point $c_{\mathcal{B}_i}$ **do**
      $S_j = \{v_j, c_{\mathcal{B}_i}\}$, set S_j as initial monitors;
      M_j = Algorithm 2 $(\mathcal{B}_i, S_j, \mathbf{P}_{\mathcal{B}_i}) - \{c_{\mathcal{B}_i}\}$;
    $M(\mathcal{B}_i,\mathbf{P}_{\mathcal{B}_i}) = \arg\min(|M_j|)$;
    $M(\mathcal{G}^t,\mathbf{P}) = M(\mathcal{G}^t,\mathbf{P}) \cup M(\mathcal{B}_i,\mathbf{P}_{\mathcal{B}_i})$;
  $\mathcal{M}(\mathcal{G}^t,\mathbf{P})$ = Algorithm 2 $(\mathcal{G}^t, M(\mathcal{G}^t,\mathbf{P}),\mathbf{P})$;
return $M(\mathcal{G},\mathbf{P}) = M(\mathcal{G}^t,\mathbf{P})$

Let $M^*()$ denote the optimal monitor placement. We have the following theorems.

Theorem 7. *For each edge-BC $\mathcal{B}$, $M^*(\mathcal{B},\mathbf{P}_{\mathcal{B}})$ belongs to one optimal monitor placement $M^*(\mathcal{G},\mathbf{P})$.*

Proof. Refer to Section 3.6.2. ■

Theorem 8. *If a network $\mathcal{G}$ and a target $\mathbf{P}$ are given, Algorithm 3 can generate an optimal monitor placement for inferring all the paths in $\mathbf{P}$, i.e., (i) all paths in $\mathbf{P}$ can be identified, and (ii) no placement can identify all the interested paths with a smaller number of monitors.*

Proof. Refer to Section 3.6.2. ■

3.5 Evaluation

We evaluate the optimal monitor placement for a given set of interest paths in real-world ISP networks collected by Rocketfuel [40]. We select four ISP networks, the details of which are given in Table 3.2, where $N_{\mathcal{B}}$ denotes the number of BCs within a network.

Table 3.2: **Characteristics of ISP Networks**

AS	ISP Name	$\|L\|$	$\|V\|$	Avg. Node Deg.	$N_{\mathcal{B}}$
AS1755	Ebone (Europe)	381	172	4.430	28
AS3967	Exodus (US)	434	201	4.318	38
AS3257	Tiscali (Europe)	404	240	3.366	142
AS7018	AT&T (US)	2078	631	6.586	58

We compare our monitor placement algorithm with OMA [14], which can identify *a set of links* with a minimum number of monitors. Given a graph and a set of interested paths, OMA takes all the links included by the interested paths as a set of given interested links, and obtains the path performance by deploying monitors to identify all those links.

In the experiment, the number of interested paths varies in the range of $[50, 100]$. We randomly generate interested paths where the path length follows uniform distribution over the range of $[1, 15]$. For each network topology and the number of interested paths, we repeat the test 50 times and report the average number as well as standard deviations.

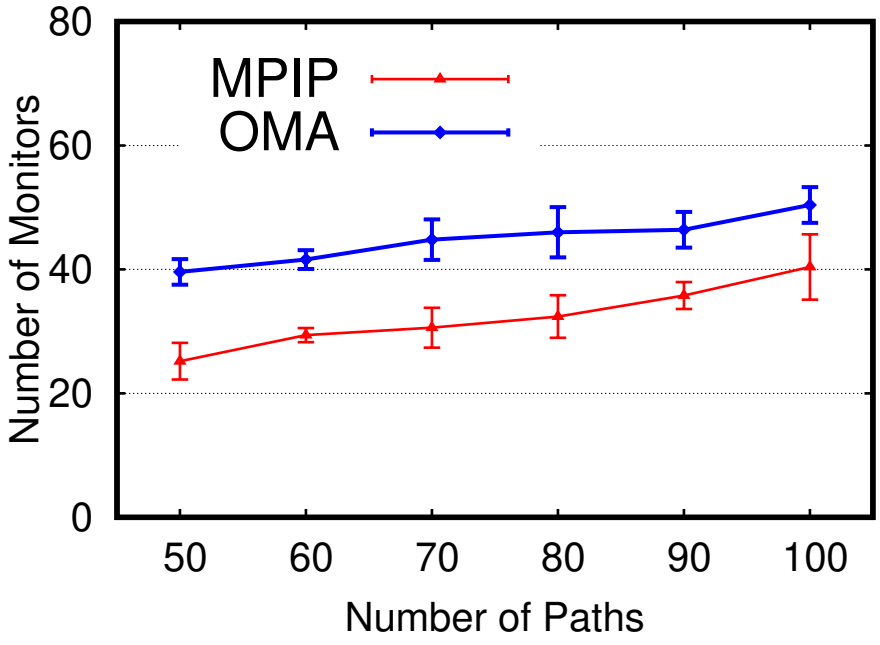


Figure 3.9: Monitor placement results in ISP network AS3967.

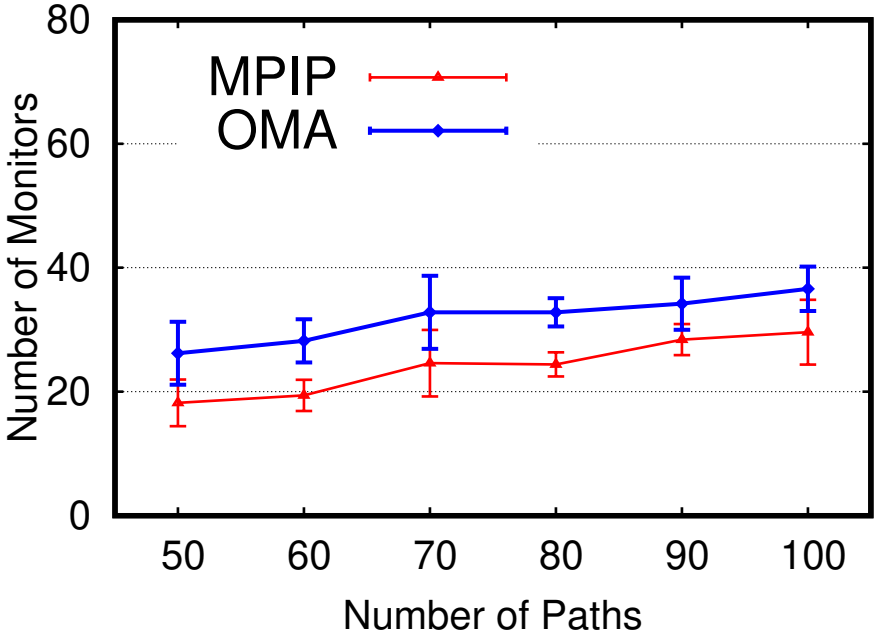


Figure 3.10: Monitor placement results in ISP network AS1755.

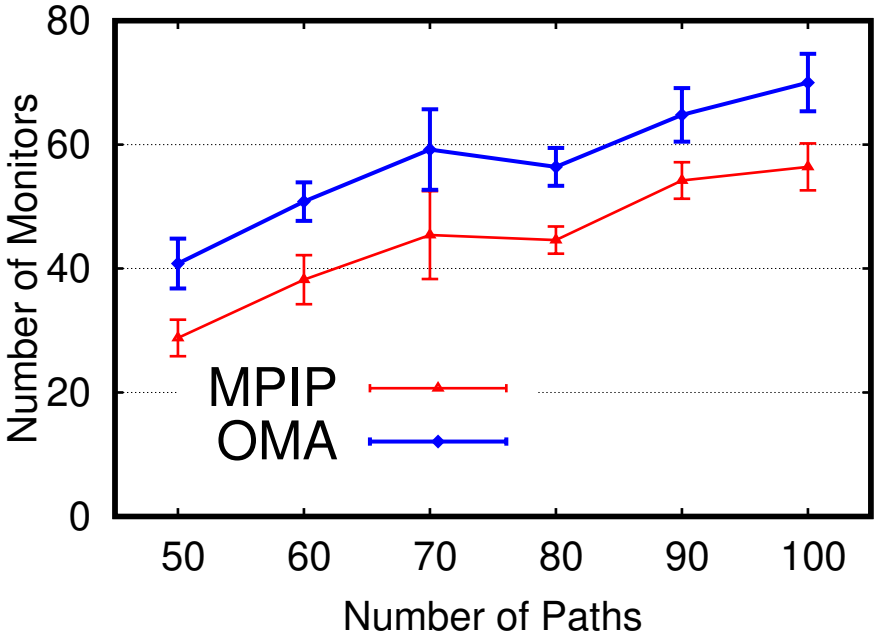


Figure 3.11: Monitor placement results in ISP network AS3257.

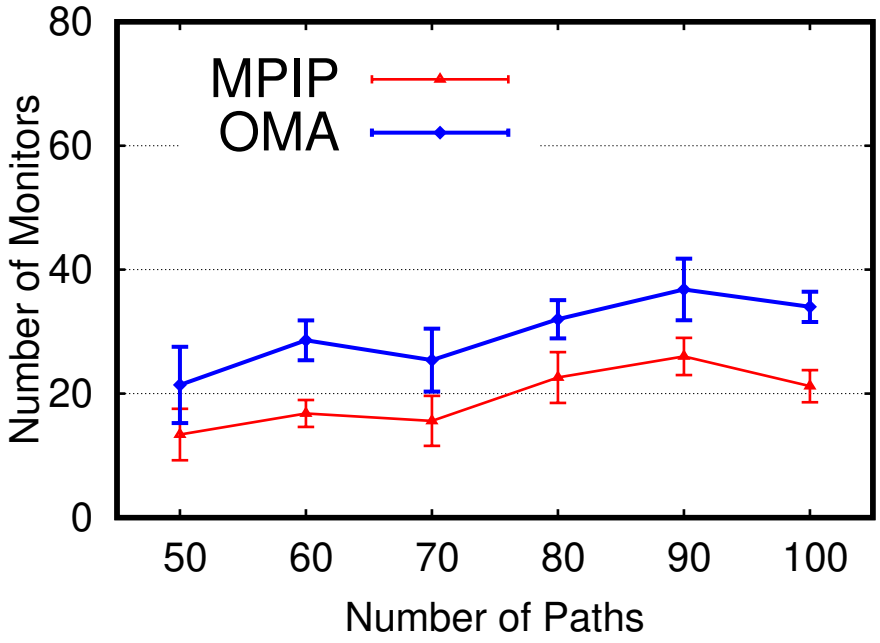


Figure 3.12: Monitor placement results in ISP network AS7018.

Table 3.3: **The number of identified links**

AS \ |P|		50	60	70	80	90	100
AS1755	OMA	255	268	303	310	314	333
	MPIP	220	232	265	276	278	295
	$\|L_u\|$	35	36	38	34	36	38
AS3967	OMA	278	291	330	340	350	370
	MPIP	232	243	286	287	293	313
	$\|L_u\|$	46	48	44	53	57	57
AS3257	OMA	229	233	261	269	280	290
	MPIP	173	183	205	214	216	226
	$\|L_u\|$	56	50	56	55	64	64
AS7018	OMA	332	360	428	464	476	541
	MPIP	313	343	407	441	449	513
	$\|L_u\|$	19	17	21	23	27	28

The results of monitor placement are shown in Fig. 3.9, Fig. 3.10, Fig. 3.11, and Fig. 3.12, where the x-axis represents the number of the interested paths and y-axis represents the number of monitors required to identify all the interested paths. When the number of interested paths

increases from 50 to 100, more monitors are needed in both our algorithm and OMA. Under all four network topologies, our MPIP algorithm uses $20\% - 41\%$ fewer monitors than the OMA algorithm. In addition, AS7081 uses the smallest number of monitors, because according to Table 3.2 it has the highest average node degree and likely better network connectivity.

The MPIP algorithm uses less number of monitors than the OMA algorithm, because solving OMP does not need to identify all links included in the set of interested paths. The OMA algorithm is an overkill, since it solves OMP by inferring the metrics of all links in the set of interested paths. Although MPIP also needs to identify some individual links, the number of individual links that MPIP needs to identify is smaller than that identified with the OMA algorithm. The results are reported in Table 3.3, where $|L_u|$ denotes the extra number of links that the OMA algorithm needs to identify.

From the table, we can see that when the number of interested paths increases, $|L_u|$ remains roughly stable in the same Topology. Nevertheless, $|L_u|$ varies widely in different networks. Checking the network characteristics in Table 3.2, we can see that AS7018 has a much higher average node degree than the others. Intuitively, a network with a higher average node degree normally has better network connectivity, and thus with a given set of monitors, the number of unidentifiable links is smaller.

3.6 Proofs

3.6.1 Fundamental Lemmas and Theorems

In this section, we give all the relevant lemmas and theorems that serve as building blocks for the lemmas and theorems in former sections.

Definition 4. *[31] A link l is called a **cross-link** with respect to two monitors m_1 and m_2 if there are four paths between m_1 and m_2, $\mathcal{P}_\mathcal{A}$, $\mathcal{P}_\mathcal{B}$, $\mathcal{P}_\mathcal{C}$, and $\mathcal{P}_\mathcal{D}$, each formed from paths $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ by*

$$\begin{cases} \mathcal{P}_\mathcal{A} = \mathcal{P}_1 \cup \mathcal{P}_2 \\ \mathcal{P}_\mathcal{B} = \mathcal{P}_3 \cup \mathcal{P}_4 \\ \mathcal{P}_\mathcal{C} = \mathcal{P}_1 \cup l \cup \mathcal{P}_4 \\ \mathcal{P}_\mathcal{D} = \mathcal{P}_3 \cup l \cup \mathcal{P}_2, \end{cases} \tag{3.1}$$

such that

$$\begin{cases} |\mathcal{P}_1 \cap \mathcal{P}_2| = 1 \\ |\mathcal{P}_3 \cap \mathcal{P}_4| = 1 \\ |\mathcal{P}_2 \cap \mathcal{P}_3| = 0 \\ |\mathcal{P}_1 \cap \mathcal{P}_4| = 0, \end{cases} \tag{3.2}$$

where $\cup$ means the concatenation of paths, $\cap$ returns the set of intersection points of two paths, and $|.|$ means the cardinality of a set.

Definition 5. *[31] A link l is called a **shortcut** if $\exists$ a simple path $\mathcal{P}_3$ whose metric has been*

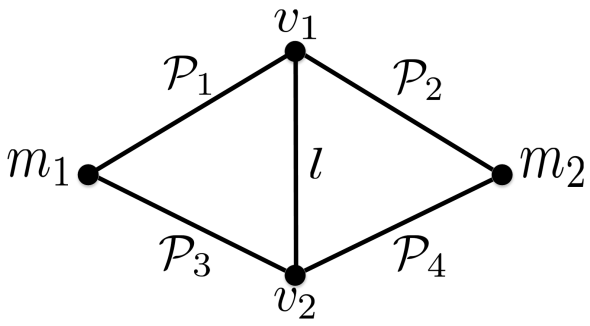


Figure 3.13: A cross-link l.

identified such that the following m_1 to m_2 simple paths can be formed:

$$\mathcal{P}_{\mathcal{A}} = \mathcal{P}_1 \cup l \cup \mathcal{P}_2 \tag{3.3}$$

$$\mathcal{P}_{\mathcal{B}} = \mathcal{P}_1 \cup \mathcal{P}_3 \cup \mathcal{P}_2, \tag{3.4}$$

satisfying $|\mathcal{P}_1 \cap \mathcal{P}_3| = 1$, $|\mathcal{P}_2 \cap \mathcal{P}_3| = 1$, and $|\mathcal{P}_1 \cap \mathcal{P}_2| = 0$.

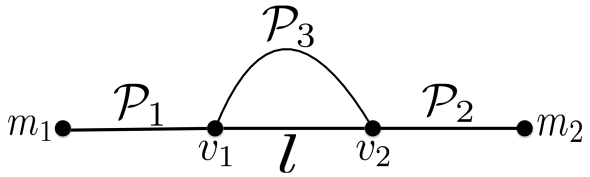


Figure 3.14: A shortcut l.

Definition 6. *A path p is called a **cross-path** with respect to two monitors m_1 and m_2 if there are four paths between m_1 and m_2, $\mathcal{P}_{\mathcal{A}}$,$\mathcal{P}_{\mathcal{B}}$,$\mathcal{P}_{\mathcal{C}}$, and $\mathcal{P}_{\mathcal{D}}$, each formed from paths $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$,$\mathcal{P}_4$ by*

$$\begin{cases} \mathcal{P}_{\mathcal{A}} = \mathcal{P}_1 \cup \mathcal{P}_2 \\ \mathcal{P}_{\mathcal{B}} = \mathcal{P}_3 \cup \mathcal{P}_4 \\ \mathcal{P}_{\mathcal{C}} = \mathcal{P}_1 \cup p \cup \mathcal{P}_4 \\ \mathcal{P}_{\mathcal{D}} = \mathcal{P}_3 \cup p \cup \mathcal{P}_2, \end{cases} \tag{3.5}$$

such that

$$\begin{cases} |\mathcal{P}_1 \cap \mathcal{P}_2| = 1 \\ |\mathcal{P}_3 \cap \mathcal{P}_4| = 1 \\ |\mathcal{P}_2 \cap \mathcal{P}_3| = 0 \\ |\mathcal{P}_1 \cap \mathcal{P}_4| = 0, \end{cases} \tag{3.6}$$

where $\cup$ means the concatenation of paths, $\cap$ returns the set of intersection points of two paths, and $|.|$ means the cardinality of a set.

Definition 7. *A path p is called a **shortcut** if $\exists$ a simple path $\mathcal{P}_3$ whose metric has been identified*

such that the following m_1 to m_2 simple paths can be formed:

$$\mathcal{P}_{\mathcal{A}} = \mathcal{P}_1 \cup p \cup \mathcal{P}_2 \tag{3.7}$$

$$\mathcal{P}_{\mathcal{B}} = \mathcal{P}_1 \cup \mathcal{P}_3 \cup \mathcal{P}_2, \tag{3.8}$$

satisfying $|\mathcal{P}_1 \cap \mathcal{P}_3| = 1$, $|\mathcal{P}_2 \cap \mathcal{P}_3| = 1$, and $|\mathcal{P}_1 \cap \mathcal{P}_2| = 0$. We also name $\mathcal{P}_3$ as ***the shortcut of*** *p (p is the shortcut of $\mathcal{P}_3$ reversely).*

Theorem 9. *A link l is identifiable by two monitors m_1 and m_2 if and only if it is a cross-link or a shortcut.*

Proof. Sufficient part.

If l is a cross-link or a shortcut, then l is identifiable according to the definition of cross-link and shortcut.

Necessary part.

The necessary condition is equivalent to the following statement, "if l is neither a cross-link nor a shortcut, then l is unidentifiable".

Conditions for l not being a shortcut: If l is not a shortcut, there does not exist a path p satisfying all of the conditions: (i) $E(p) = E(l)$, (ii) $l \nsubseteq L(p)$ and (iii) p is identifiable.

Conditions for l not being a cross-link: According to Definition 4, a cross-link l must meet the conditions in (3.1), i.e., there exist four base paths $\{\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3, \mathcal{P}_4\}$, denoted as *path-requirement*, and the conditions in (3.2), i.e., the four base paths must satisfy the restrictions on their vertex, denoted as *vertex-requirement*. Hence, if l is not a cross-link, it must violate either *path-requirement* or *vertex-requirement*.

We thus can prove the necessary condition as follows. We first prove that if l is not a shortcut and it l violates conditions in *path-requirement*, then l is unidentifiable. We then prove if l is not a shortcut and l violates conditions in *vertex-requirement*, l is also unidentifiable.

1) *Violating path-requirement*

In $\mathcal{G}$, if at least three paths of $\{\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3, \mathcal{P}_4\}$ are missing, as shown in Fig. 3.15, then at least one monitor (e.g., m_2) cannot be connected to the end nodes of l. However, such situation cannot exist since $\mathcal{G}$ is a connected graph.

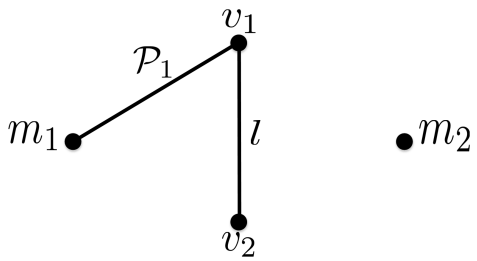


Figure 3.15: violating connectivity case.

Then, as illustrated in Fig. 3.16, to violate *path-requirement*, no more than three paths of $\{\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3, \mathcal{P}_4\}$ could be missing, which leads to the following four cases:

(a1) There is only one missing path (e.g., $\mathcal{P}_1$ in case (a1) of Fig. 3.16);

(a2) There are two missing paths, which are connected to the same monitor (e.g., $\mathcal{P}_1$ and $\mathcal{P}_3$ in case (a2) of Fig. 3.16). **This case cannot hold** since $\mathcal{G}$ is connected. Thus we focus on proving that l is not identifiable in rest cases.

(a3) There are two missing paths, which are connected to different monitors and different end nodes of l (e.g., $\mathcal{P}_1$ and $\mathcal{P}_4$ in case (a3) of Fig. 3.16);

(a4) There are two missing paths, which are connected to different monitors but the same end node of l (e.g., $\mathcal{P}_3$ and $\mathcal{P}_4$ in case (a4) of Fig. 3.16).

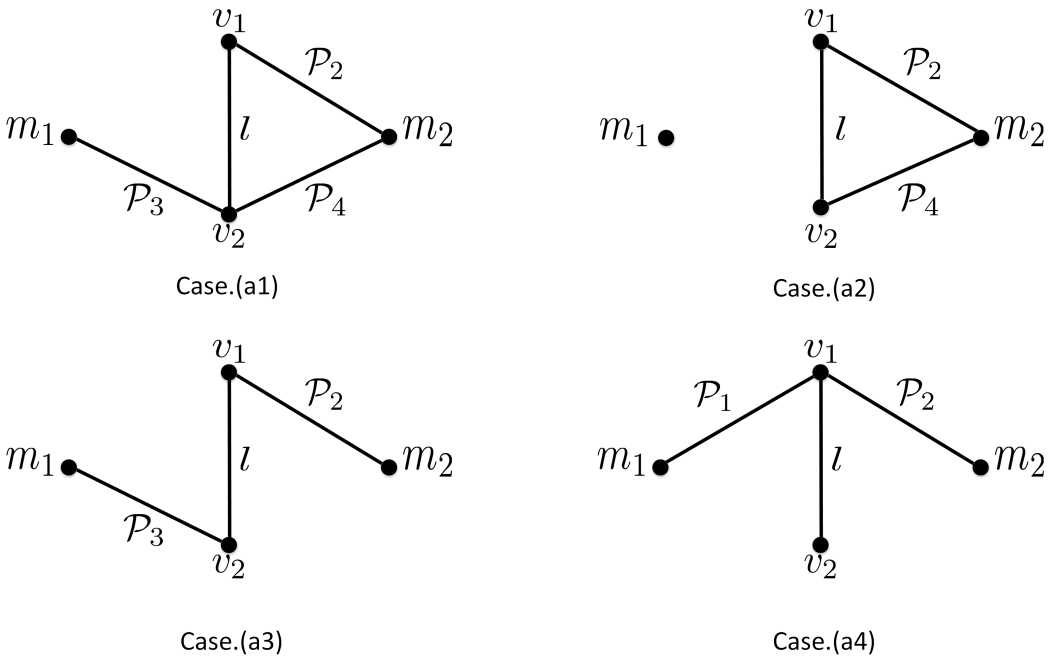


Figure 3.16: Four cases violating path-requirement.

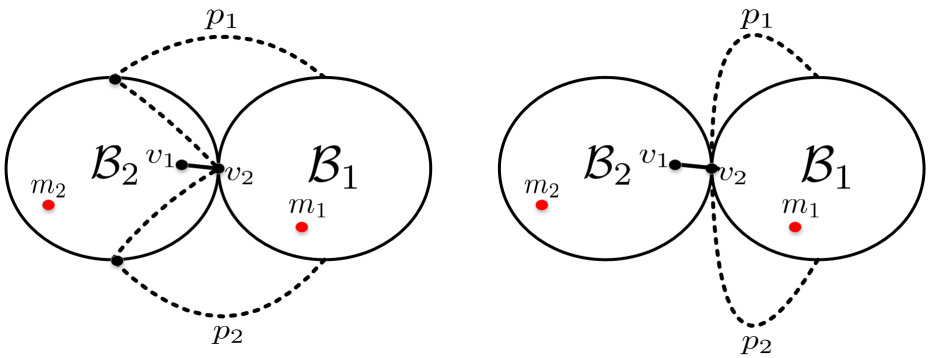


Figure 3.17: For all measurement paths that contain l, v_2 acts as a cut-point.

In case (a1), any path from m_1 to v_1 must go through v_2, then m_1 cannot be connected to v_1 when v_2 is removed. Hence, v_2 is a cut-point for all measurement paths that contain l in $\mathcal{G}$, as illustrated in Fig. 3.17. In addition, l is incident to v_2.

To identify l, we need to consider the corresponding linear system formed by measurement paths. The measurement paths used to identify l can be divided into 2 groups: one including the measurement paths that contain l, and the other including the measurement paths that do not contain l.

The measurement paths in the first group correspond to the linear system

$$R_1 w = c_1. \tag{3.9}$$

The measurement paths in the second group correspond to the linear system

$$R_2 w = c_2. \tag{3.10}$$

Note that R_1, R_2 are a Boolean matrix in the form of R_{ij}, with each entry R_{ij} denoting whether link l_j is present on a measurement path in group one (R_1) or in group two (R_2); w is a column vector of all link metrics; c_1 is the column vector of all path measurements in group one, and c_2 is the column vector of all path measurements in group two.

We in the rest prove that if we only consider linear system (3.9), l is unidentifiable, and combining linear system (3.9) with (3.10) will not bring effective information for identifying l.

In (3.9), since for all corresponding measurement paths that contain l, v_2 acts as a cut-point, and l is a link incident to this cut-point. We prove the unidentifiability of l by making the strongest assumption: for any measurement path that contains $l(v_1v_2)$, the segment between v_2 and m_1 is identifiable, i.e., as illustrated in Fig. 3.18, the part of any measurement path in $\mathcal{B}_1$ is identifiable. If l is not identifiable with the strongest assumption, l is also not identifiable with any weaker assumptions, because a weaker assumption means that the part of some measurement paths in $\mathcal{B}_1$ is unidentifiable.

With the strongest assumption, we can treat the cut-point v_2 as a monitor. Since any part of the measurement path in $\mathcal{B}_1$ is identifiable, deducting their metrics in the existing linear equation gives new linear equation corresponding to new measurement path that starts from m_2 to v_2.

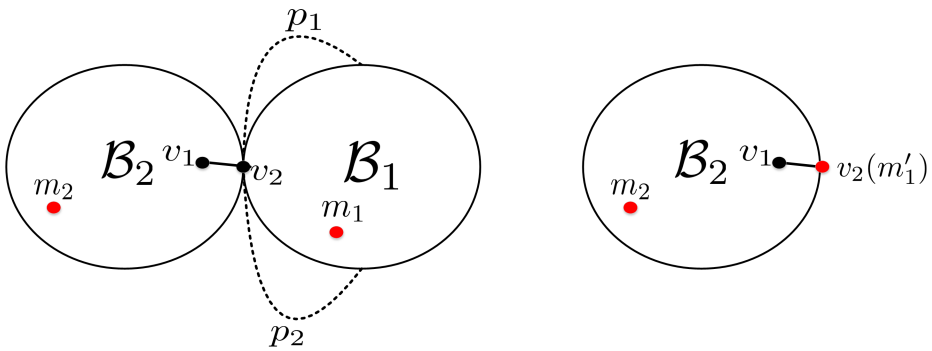


Figure 3.18: If any measurement path in $\mathcal{B}_1$ is identifiable, v_2 can be treated like a monitor in $\mathcal{B}_2$.

Under such condition, l is equivalent to an exterior link (link incident to a monitor), as shown in Fig. 3.18. For an exterior link, *Theorem III.1* in [31] has showed that it is unidentifiable. Hence, if we only consider the linear system (3.9), even with the strongest assumption, l is unidentifiable;

therefore, l cannot be identified only by (3.9).

We next show that adding equations in (3.10) to (3.9) would not change the unidentifiability of l. In the proof of *Theorem III.1* in [31], it has been showed that even if the interior graph of $\mathcal{B}_2$ (refer to Fig. 3.18) is identifiable, we can at most get a set of equations which assume the following form

$$W_{a_i} + W_{b_j} = c_{ij}$$

$$i = 1, \cdots, k_1, \quad j = 1, \cdots, k_2$$

where W_{a_i} represents the metric of the exterior link that is incident to m_1' and W_{b_j} represents the metric of the exterior link that is incident to m_2, c_{ij} are known values. The above equations are linearly independent [31]. Now l is one of the links in a_i, $i = 1, \cdots, k_1$, and the metric of l is denoted as W_l. In order to combine equations in (3.10) to obtain W_l, we must know at least one W_{b_j}, $j = 1, \cdots, k_2$ from (3.10), which is impossible because every b_j is an exterior link that is incident to m_2 and an exterior link is always unidentifiable.

To conclude, l is unidentifiable in case (a1).

In case (a3), any path from $m_1(m_2)$ to $v_1(v_2)$ must go through $v_2(v_1)$, then *l is a bridge for all measurement paths containing l.* Therefore, when we only consider the system of linear equations formed by the measurement paths containing l, we can use the following linear equations

$$Rw = c$$

to denote the case, where R, w, c have the same meaning as in (3.9). According to *Lemma A.1* in [31], l is unidentifiable since it is a bridge.

There may exist other measurement paths that do not contain l but may help identify l. Correspondingly, we denote these measurement paths with linear system

$$R'w = c'.$$

Similar to case (a1), we show that the unidentifiability of l cannot be changed by this part of linear equations.

Lemma A.1 in [31] has proved that "a bridge in $\mathcal{G}$ with one monitor on each side, then neither l nor its adjacent links are identifiable", the proof of this lemma showed that for a bridge l, even with the strongest assumption (i.e., all links except l and its adjacent links are identifiable), the reduced linear equation associated with any $m_1 \rightarrow m_2$ path assumes the form

$$W_{a_i} + W_l + W_{b_j} = c_{ij}$$

$$i = 1, \cdots, k_1, \quad j = 1, \cdots, k_2$$

where W_{a_i} and W_{b_j} represent the metrics of l's adjacent links on each side, W_l is the metric of l, and c_{ij} are known values. These equations are linearly independent [31]. Now if we want to use $R'w = c'$ to change the unidentifiability of l in $Rw = c$, we must at least know one W_{a_i} and one W_{b_j} from $R'w = c'$. However, i), we cannot obtain the metric of a_i or b_j independently, using the same

argument in case (a1), i.e., a_i or b_j is a link that is incident to a cut-point for all measurement paths that contains itself (the two vertices of a bridge could be seen as two cut-points described in case (a1). Here, v_1 and v_2 are the two cut-points); ii), we cannot get the combined metric of a_i and b_j, i.e., the sum of W_{a_i} and W_{b_j}, from $R'w = c'$, because otherwise l becomes a shortcut, contradicting the assumption.

In case (a4), since any path from m_1 or m_2 to v_2 must go through v_1, any measurement path containing l cannot avoid a loop $v_1 \to v_2 \to v_1$. This contradicts the assumption that the measurement path is cycle-free.

2) *Violating vertex-requirement*

Now we prove even if l satisfies conditions in *path-requirement*, violating conditions in *vertex-requirement* makes l unidentifiable.

By symmetry in (3.2), we can just consider two cases that violate the *vertex-requirement*,

(1) $|\mathcal{P}_1 \cap \mathcal{P}_2| \neq 1$;

(2) $|\mathcal{P}_2 \cap \mathcal{P}_3| \neq 0$.

Firstly, $|\mathcal{P}_1 \cap \mathcal{P}_2| \neq 1$ means either $|\mathcal{P}_1 \cap \mathcal{P}_2| = 0$ or $|\mathcal{P}_1 \cap \mathcal{P}_2| > 1$. Since the satisfied *path-requirement* guarantees $|\mathcal{P}_1 \cap \mathcal{P}_2| \neq 0$, we must have $|\mathcal{P}_1 \cap \mathcal{P}_2| > 1$, that is, there must be a vertex v_3, which any path from $m_1(m_2)$ to v_1 must go through if v_2 is removed, as shown in Fig. 3.19. In such a case, l and v_1v_3 form a 2-bridge-cut, which is unidentifiable with 2 monitors [31]. Thus l in this 2-bridge-cut is unidentifiable.

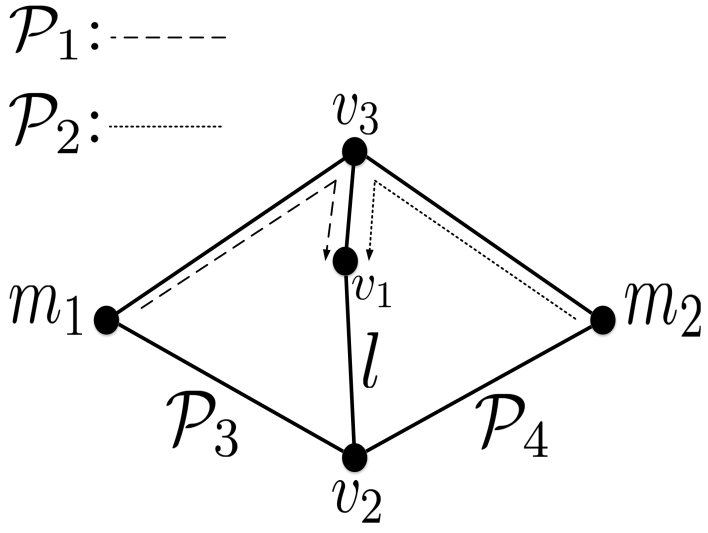


Figure 3.19: m_1v_3 and m_2v_3 represent all paths between m_1 and v_3 and all paths between m_1 and v_3, respectively.

Secondly, $|\mathcal{P}_2 \cap \mathcal{P}_3| \neq 0$ means $\mathcal{P}_2$ at least intersects $\mathcal{P}_3$ at one node. By symmetry, there are three sub-cases.

- $\mathcal{P}_3$ intersects $\mathcal{P}_2$ at m_2. In this sub-case, any path from m_1 to v_2 must go through m_2 as shown in Fig. 3.20. Since we do not allow measurement path to go through 2 monitors and

then go back to the interior graph of $\mathcal{G}$, the existence of such $\mathcal{P}_3$ cannot be contained in any measurement path. In this case, $\mathcal{P}_3$ can be ignored since it does not help identify l. The remained part is equivalent to case (a1), where we have proved that l is unidentifiable.

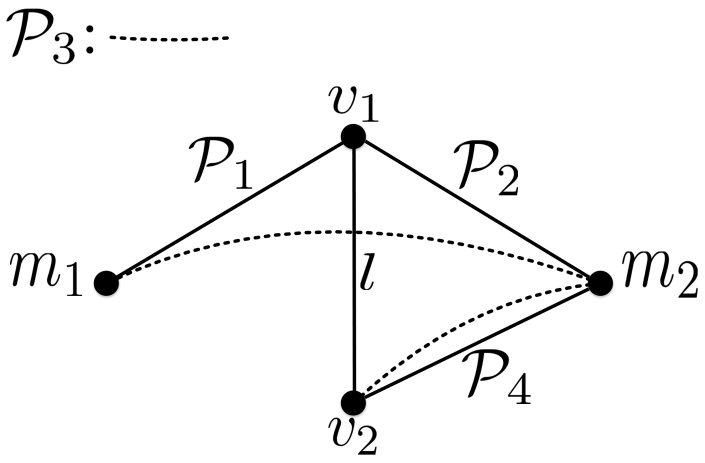


Figure 3.20: $\mathcal{P}_3$ intersects $\mathcal{P}_2$ at m_2.

- $\mathcal{P}_3$ intersects $\mathcal{P}_2$ at v_1. In this sub-case, any path from m_1 to v_2 must go through v_1 as shown in Fig. 3.21. This sub-case is essentially the same as case (a1), where the unidentifiability of l is proved.

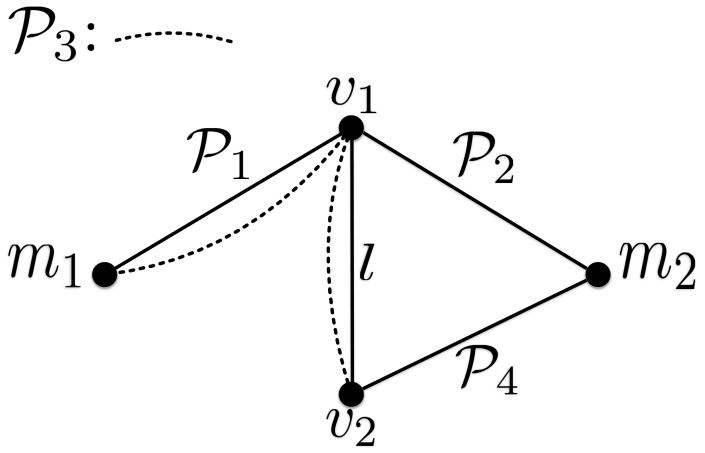


Figure 3.21: $\mathcal{P}_3$ intersects $\mathcal{P}_2$ at v_1.

- $\mathcal{P}_3$ intersects $\mathcal{P}_2$ at (one or more) nodes except v_1 and m_2. This sub-case includes two scenarios, depicted in Fig. 3.22.

    1. In scenario (A), the neighboring vertex of v_1 and v_2 are the same vertex in the intersected part of $\mathcal{P}_2$ and $\mathcal{P}_3$, as illustrated in Fig. 3.22 (A). Now link $l_{v_1,a}$ and $l_{v_2,a}$ are 2 cross-links which are identifiable, thus path $p_{v_1av_2}$ is identifiable, making l a shortcut. This contradicts the assumption that l is not a shortcut.

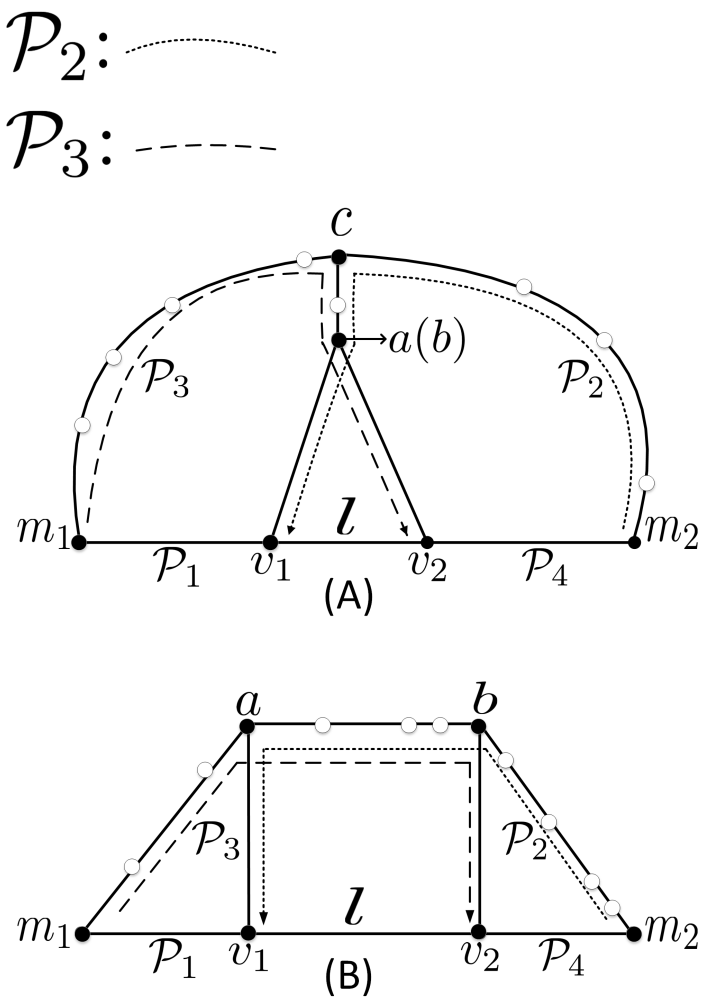


Figure 3.22: Two intersection scenarios of $\mathcal{P}_2$ and $\mathcal{P}_3$.

2. In scenario (B), the neighboring vertices of v_1 and v_2 are different two vertices in the intersected part of $\mathcal{P}_2$ and $\mathcal{P}_3$. As shown in Fig. 3.22 (B), in the intersected part of $\mathcal{P}_2$ and $\mathcal{P}_3$, the neighboring vertex of v_1 is a and the neighboring vertex of v_2 is b.

We now prove that l is unidentifiable in scenario (B), which can be further classified into the following categories.

i) *l with a link in path p_{ab} forms a 2-bridge-cut.*

Without loss of generality, we assume that l_{av_3} together with l forms the 2-bridge-cut, as shown in Fig. 3.23 (B-1). Then l is unidentifiable as a 2-bridge-cut is always unidentifiable with 2 monitors [31].

ii) *l with any link in path p_{ab} does not forms a 2-bridge-cut.* Without loss of generality, we consider when l_{av_3} together with l does not form the 2-bridge-cut. In such case, there exists *at least* one of two paths $\{p_{av_2}, p_{v_3v_1}\}$, where p_{av_2} is a path connecting a and v_2 without

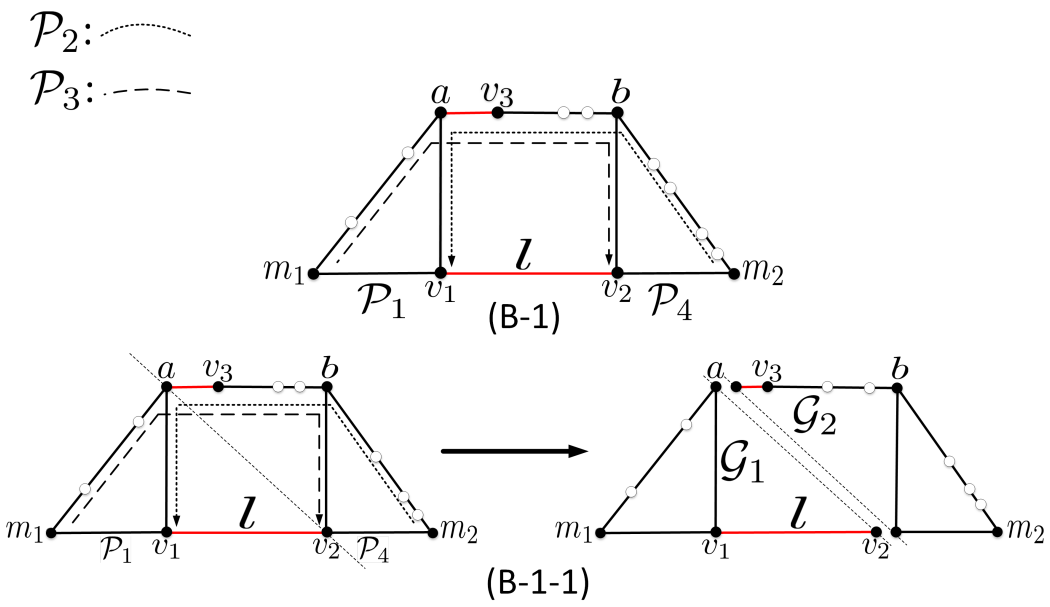


Figure 3.23: l with l_{av_3} forms a 2-bridge-cut.

going through b and v_1, $p_{v_3v_1}$ is a path connecting v_3 and v_1 without going through a and v_2 (otherwise l with l_{av_3} forms a 2-bridge-cut).

Assume that p_{av_2} exists, then we take it into account.

If p_{av_2} is disjoint to the four base paths ($\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$), as shown in Fig. 3.24, then p_{av_2} is a cross-path, and with cross-link l_{a,v_1}, path $p_{v_1av_2}$ becomes identifiable, making l become a shortcut. This contradicts the assumption that "l is neither a cross-link nor shortcut".

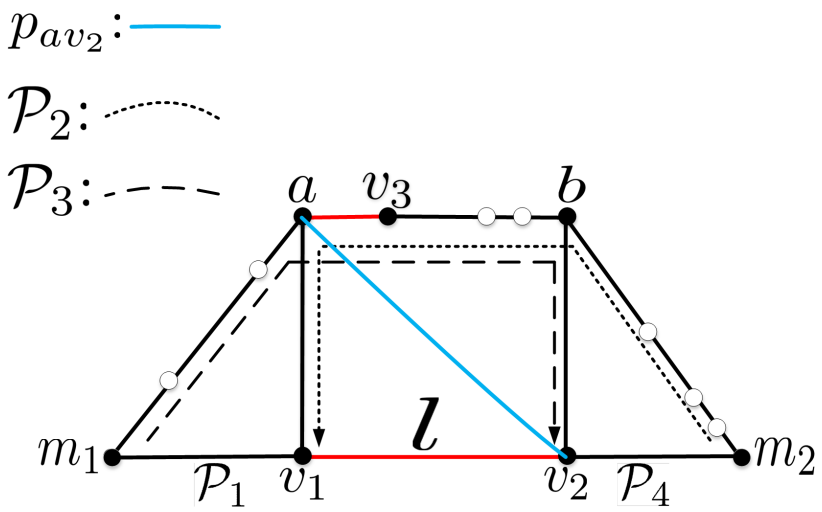


Figure 3.24: p_{av_2} is disjoint to the four base paths ($\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$).

If p_{av_2} is not disjoint to the four base paths ($\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$), we prove that once p_{av_2} intersects $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$, respectively, l is either a cross-link or a shortcut, leading to a contradiction.

1) p_{av_2} intersects $\mathcal{P}_1$, as shown in Fig. 3.25 (B-2-1). In this case, l is a cross-link, where the four base paths ($\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$) are $p_{m_1v_4v_1}$, $p_{m_1v_4v_2}$, $p_{m_2bv_3av_1}$, $p_{m_2v_2}$.
2) p_{av_2} intersects $\mathcal{P}_2$, as shown in Fig. 3.25 (B-2-2). In this case, l is a shortcut. Indeed, l_{av_1}, l_{bv_2} are 2 cross-links and p_{ab} is a cross-path, which are identifiable, thus $p_{v_1av_3bv_2}$ is identifiable, making l a shortcut.
3) p_{av_2} intersects $\mathcal{P}_3$, as shown in Fig. 3.25 (B-2-3). l is a cross-link, where the four base paths ($\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$) are $p_{m_1v_1}$, $p_{m_1v_4v_2}$, $p_{m_2v_2}$, $p_{m_2bv_3av_1}$.
4) p_{av_2} intersects $\mathcal{P}_4$, as shown in Fig. 3.25 (B-2-4). l_{av_1}, l_{bv_2} are 2 cross-links and p_{ab} is a cross-path, which are identifiable, thus $p_{v_1av_3bv_2}$ is identifiable, making l a shortcut.

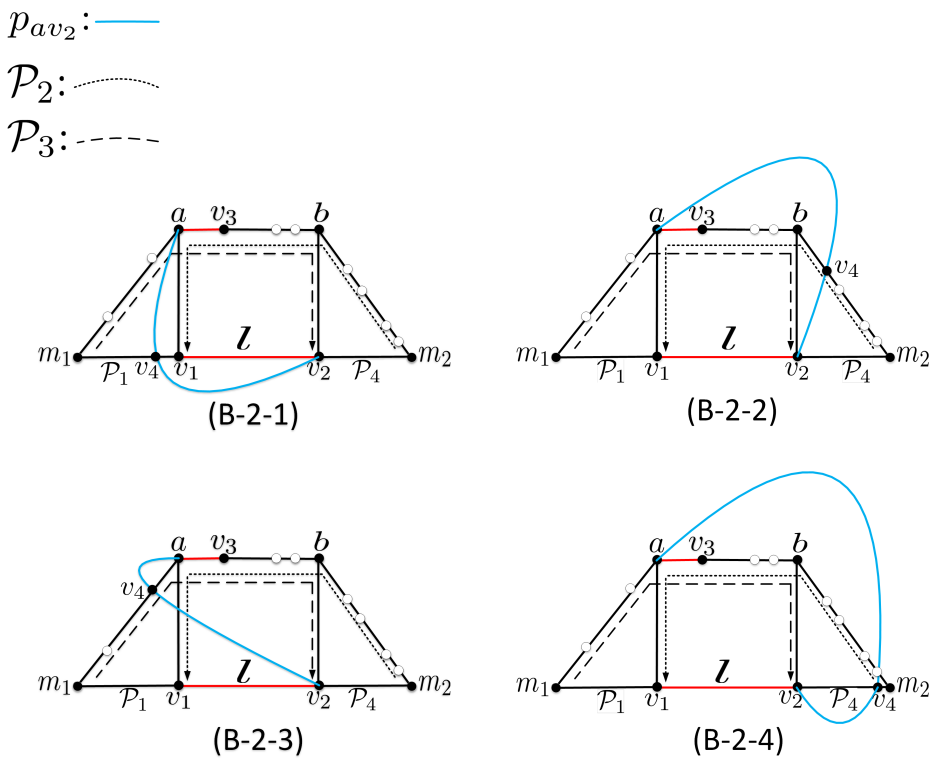


Figure 3.25: Four cases of p_{av_2} is not disjoint to the four base paths ($\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$).

■

Theorem 10. *With respect to 2 monitors, if p is contained in at least one measurement path, then the sufficient and necessary condition for it to be identifiable is that p is a cross-path or a shortcut.*

Proof. Sufficient part.

If p is a cross-path or a shortcut, then p is identifiable according to the definition of cross-path and shortcut.

Necessary part.

We need to prove that "if p is contained on at least one measurement path and p is neither a cross-path nor a shortcut, then p is unidentifiable".

Firstly, we treat p as a whole piece, i.e., a virtual link. With Theorem 9, the metric of p, denoted as W_p, cannot be uniquely determined, because p as a virtual link is not a cross-link or a shortcut.

Therefore, we need to consider measurement paths that go through inner vertex of p (vertex of p except two end nodes), because otherwise the measurement paths cannot help identify p. If p can be uniquely determined, such identifiability must be achieved by the identifiable segments that compose p, because p is neither a cross-path nor a shortcut, excluding the case that it can be identified as a whole piece—a virtual link.

With the above analysis, there are only two ways to identify p:

1. "On its own" as a virtual link, p is a cross-path or a shortcut. This cannot be true, because it contradicts the assumption that "p is neither a cross-path nor a shortcut".

2. By two identifiable segments that compose p. Note that if there are 3 or more identifiable segments such that $p = s_1 + s_2 + s_3$, we can always recursively reduce to the case of two segments by $p = s'_1 + s'_2$, $s'_1 = s_1 + s_2$ and $s'_2 = s_3$.

We next prove that case 2) is also impossible by showing that one of the two segments of p must be unidentifiable.

Similar to the proof of Theorem 9, we study the situations where p is not a cross-path, from the conditions for *path-requirement* as well as *vertex-requirement*.

Recall that in the the proof of Theorem 9, from the *path-requirement*, there are 4 cases that prevent l from being a cross-link. But for a path p, there are only 2 cases in which p is not a cross-path from the *path-requirement*, as shown in Fig. 3.26. This difference is due to the basic assumption that "p is on at least one measurement path" excludes case (a2) and case (a4) in Fig. 3.16.

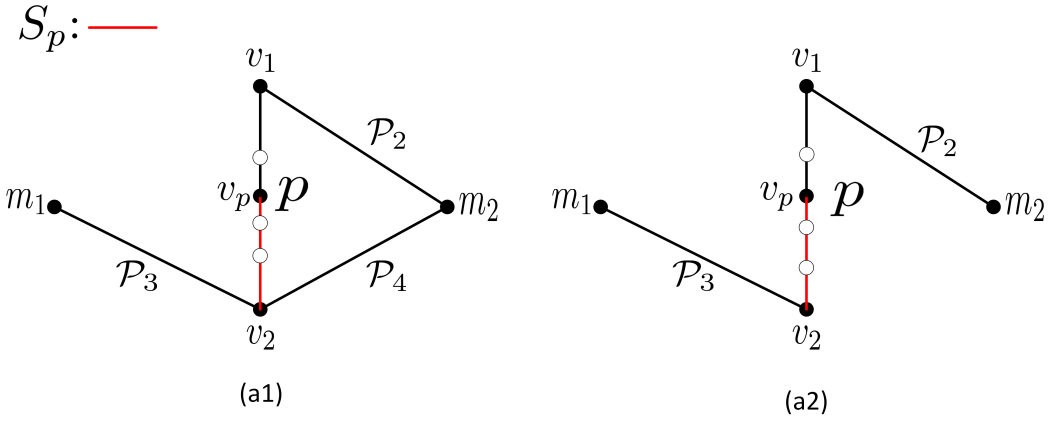


Figure 3.26: Two cases for p not being a cross-path from the *path-requirement*.

Specifically, we prove the unidentifiability of p in Fig. 3.26 (a1), and the proof of the other case is similar. Since p is neither a cross-path nor a shortcut, it can only be identifiable through 2 identifiable segments that compose it. Note that in Fig. 3.26 (a1), any combination of 2 segments that compose p has one segment that is incident to v_2, denoted as S_p, and the other one is denoted as S_n. We prove that every S_p is unidentifiable, causing p to be unidentifiable.

The proof that S_p is unidentifiable is as follows

S_p cannot be identified as a whole piece. Firstly, S_p cannot be a cross-path, because in (a1) of Fig. 3.26, "any path from m_1 to v_1 must go through v_2". If $S_p = p_{v_p v_2}$ is a cross-path, there will be a path $p_{m_1 v_p v_1}$ that starts from m_1 to v_1 without going through v_2, bringing our a path from m_1 to v_1 without going through v_2, violating the topology description for case (a1).

Secondly, if S_p still can be identified as a virtual link, S_p must be a shortcut, which leads to a $S_p^{'}$ of known metric that shares the end nodes of S_p. Such $S_p^{'}$ still cannot be a cross-path by the same reason as S_p. Therefore, the identifiability of $S_p^{'}$ can only be achieved by one of the following conditions:

1. it is an another shortcut.

2. it has 2 identifiable segments that compose it.

If we also consider $S_p^{'}$ as a whole piece, then $S_p^{'}$ must be a shortcut, leading to another path $S_p^{1'}$ that will fall into the same analysis procedure, recursively. If we consider each of these shortcuts as a whole piece, we will conduct an infinite series of paths $\{S_p^{n'}\}$ that all are shortcuts. This is impossible by the fact that $\mathcal{G}$ is of finite structure. Therefore, either we can conclude that S_p is unidentifiable, or there must be at least one shortcut $S_p^{k'}$ from $\{S_p^{n'}\}$, whose identifiability is given by the the second way, i.e., it has 2 identifiable segments that compose it.

We can prove that any $S_p^{k'}$ cannot achieve identifiability by the 2 identifiable segments that compose it by the same recursive analysis. For any combination of 2 segments that compose $S_p^{k'}$, there must be one that is incident to v_2, denoted as s_p, and the other segment is s_n. To analyze the identifiability s_p, it will go through the same procedure of identifiability analysis as S_p and end up concluding either s_p is unidentifiable or there is at least one shortcut $s_p^{k'}$ from the conducted series of shortcuts $\{s_p^{n'}\}$ is identifiable through its segments. The segment of $s_p^{k'}$ will go through the same analysis. Recursively, we will reach a link that cannot be cut into segments, which is exactly the link that is incident to v_2. And this link can only achieve identifiability by being a cross-link, leading to a contradiction since it cannot be a cross-link by the same reason that S_p cannot be a cross-path. Hence, the unidentifiability of every S_p, i.e., the segment of p that is incident to v_2 cannot be identified.

So far, we have proved that if p violates the *path-requirement*, it is unidentifiable. As for the *vertex-requirement*, same to the proof of Theorem 9, there are 2 cases that violate the *vertex-requirement*:

1. $|\mathcal{P}_1 \cap \mathcal{P}_2| \neq 1$;

2. $|\mathcal{P}_2 \cap \mathcal{P}_3| \neq 0$.

For 1), $|\mathcal{P}_1 \cap \mathcal{P}_2| \neq 1$ means either $|\mathcal{P}_1 \cap \mathcal{P}_2| = 0$ or $|\mathcal{P}_1 \cap \mathcal{P}_2| > 1$. Since the satisfied *path-requirement* guarantees $|\mathcal{P}_1 \cap \mathcal{P}_2| \neq 0$, we must have $|\mathcal{P}_1 \cap \mathcal{P}_2| > 1$, that is, there must be a vertex v_3, which any path from $m_1(m_2)$ to v_1 must go through if v_2 is removed. Such v_3 is either the inner vertex of p or outside p, as shown in Fig. 3.27.

If v_3 is not a inner vertex of p. Firstly, as a virtual link, p cannot be identified, because it forms a 2-bridge-cut with $l_{v_1 v_3}$. Secondly, the unidentifiability of p cannot be compensated by its segments because any segment of p that is incident to v_1 will always form a 2-bridge-cut with $l_{v_1 v_3}$, otherwise

there will be a path can get to v_1 if v_2 is removed, violating the topology description for such case. With similar argument in *path-requirement*, p is unidentifiable.

If v_3 is a inner vertex of p. Without loss of generality, we assume such v_3 is the neighboring vertex of v_1. Since any path from $m_1(m_2)$ to v_1 must go through v_3, making v_1v_3 a bridge, which is always unidentifiable [31]. With similar argument in *path-requirement*, v_1v_3 being unidentifiable cause p to be unidentifiable.

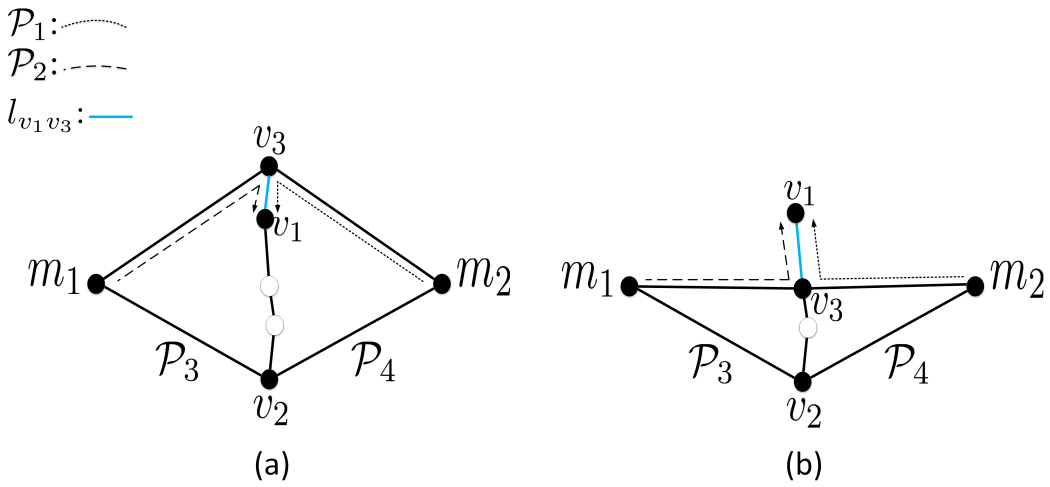


Figure 3.27: m_1v_3 and m_2v_3 represent all paths between m_1 and v_3 and all paths between m_1 and v_3, respectively.

For 2), $|\mathcal{P}_2 \cap \mathcal{P}_3| \neq 0$ means $\mathcal{P}_2$ at least intersects $\mathcal{P}_3$ at one node. By the same argument in the proof of Theorem 9 for this case, we can deduct that p is either a cross-path or a shortcut, contradicting the basic assumption.

■

Lemma 4. *In a 2-vantage TC $\mathcal{T}$, which contains no monitor, if a path p occupies only one of the vantages, then the segment of p inside this TC, denoted as $p_{\mathcal{T}}$, is unidentifiable.*

Proof. We firstly name the two vantages of $\mathcal{T}$ as μ_1 and μ_2 and assume p is only incident to μ_1.

$\mathcal{T}$ is a 2-vantage TC with no monitor in it, so to measure any path or link in $\mathcal{T}$, every cycle-free measurement path must go in $\mathcal{T}$ through one of the vantages and go back to a monitor through the other vantage. The whole graph $\mathcal{G}$ could be divided into 2 sub-graph by μ_1 and μ_2, one of which has no monitors, denoted as $\mathcal{G}_{nm}$, and the other one has monitor, denoted as $\mathcal{G}_m$, as illustrated in Fig. 3.28.

Now we prove the unidentifiability of $p_{\mathcal{T}}$ even under the ideal strongest assumption: every link in $\mathcal{G}_m$ is identifiable. With such strongest assumption, the 2 vantages, μ_1 and μ_2, act like 2 monitors, since for any MP p goes in $\mathcal{T}$ and back to a monitor, its segment s in $\mathcal{G}_m$ is identifiable, deducting the metric of s from the linear equation given by p and its measurement value, a new linear equation corresponding to new measurement path always starts from one of the vantage and ends at the other vantage of $\mathcal{T}$. This makes any $p_{\mathcal{T}}$ equivalent to an exterior path (a path that is only incident one monitor), by Lemma 7, $p_{\mathcal{T}}$ is unidentifiable. Therefore, even with the strongest assumption, $p_{\mathcal{T}}$ is

unidentifiable, $p_{\mathcal{T}}$ would not be able to be identifiable in any weaker situations, because a weaker assumption means that there are cases where the segments of p in $\mathcal{G}_m$ is unidentifiable.

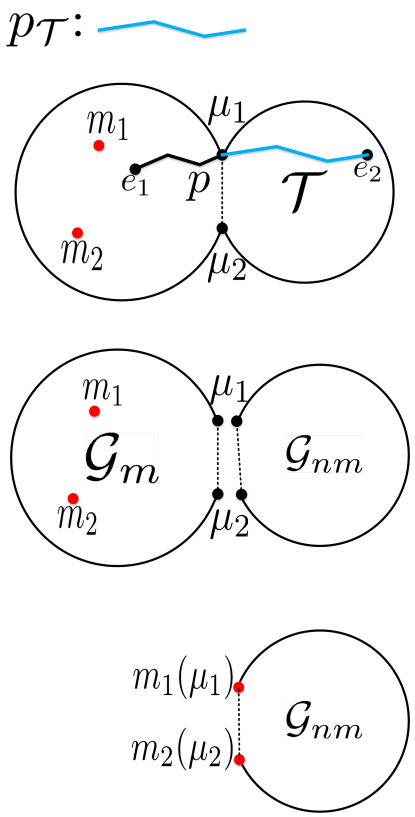


Figure 3.28: A path crosses only one vantage.

■

Lemma 5. *For $\mathcal{G}_{new}$, all links in $\mathcal{H}(\mathcal{T}_0)$ are identifiable.*

Proof. Given a graph $\mathcal{G}$ and at least two monitors, $\mathcal{G}_{new}$ of $\mathcal{G}$ can be obtained and it includes two virtual monitors m_1' and m_2'. Based on the number of the real monitors, there are two cases as follows:

1) If $\mathcal{G}$ includes only two monitors m_1 and m_2, then $\mathcal{T}_0$ includes only m_1, m_2, m_1' and m_2', since neither m_1' nor m_2' can connect to the nodes in $\mathcal{G}$ without m_1 and m_2, i.e., $\{m_1, m_2\}$ is a 2-vertex-cut in $\mathcal{G}_{new}$. Therefore, $\mathcal{H}(\mathcal{T}_0)$ in cludes only one link l_{m_1,m_2}, which is a cross link that is identifiable.

2) If $\mathcal{G}$ includes at least three monitors, which will be connected to m_1' and m_2' by virtual links in $\mathcal{G}_{new}$, then m_1' can always be connected to m_2' by removing any two nodes in $\mathcal{H}(\mathcal{T})$. Therefore, $\widetilde{\mathcal{T}}$ is 3-vertex-connected, and all the links in $\mathcal{H}(\mathcal{T})$ are identifiable.

Thus, all links in $\mathcal{H}(\mathcal{T}_0)$ are identifiable. ■

Lemma 6. *Given a graph $\mathcal{G}$ containing k monitors ($k \geq 2$), a node $v \in V(\mathcal{T}_0)$ of $\mathcal{G}_{new}$, if and only if when any two nodes of $\mathcal{G}$ have been removed, v is a real monitor or is connected to at least one real monitor.*

Proof. Let m'_1 and m'_2 denote the two virtual monitors of $\mathcal{G}_{new}$, and $\{m_k\}$ denotes the given (real) monitors in $\mathcal{G}$. $\mathcal{T}_0$ includes all real monitors in $\mathcal{G}$, since $\mathcal{T}_0$ is the TC including m'_1 and m'_2 of $\mathcal{G}_{new}$, which are only connected to real monitors in $\mathcal{G}$. Then this proof includes the following two parts.

Necessary part.

We prove that if $v \in V(\mathcal{T}_0)$, then v is a real monitor or is connected to at least one real monitor when any two nodes of $\mathcal{G}$ have been removed.

1) When $k = 2$, m'_1 and m'_2 are separated from $\mathcal{G}_{new}$ by removing m_1 and m_2, since there does not exist any monitors in the remaining nodes. Thus, $\mathcal{T}_0$ only includes two nodes included in $\mathcal{G}$, which are the real monitors m_1 and m_2.

2) When $k \geq 3$, if $v \in V(\mathcal{T}_0)$, then v is a real monitor or a node connected to all the real monitors. Then if any two nodes u_1 and u_2 of $\mathcal{G}$ have been removed, even if $\{u_1, u_2\} \subseteq V(\mathcal{T}_0)$, $\mathcal{T}_0$ is connected since it is 3-vertex-connected. Thus, the remaining node in $\mathcal{T}_0$ is a real monitor or is connected to at least one monitor, because $k \geq 3$, i.e., there is at least one monitor remained.

Therefore, in both cases, if $v \in V(\mathcal{T}_0)$, then v is a real monitor or is connected to at least one real monitor when any two nodes of $\mathcal{G}$ have been removed.

Sufficient part.

We prove that if v is a real monitor or connected to at least one real monitors when any two nodes of $\mathcal{G}$ have been removed, then $v \in V(\mathcal{T}_0)$.

Since $\mathcal{T}_0$ includes all real monitors, $v \in (\mathcal{T}_0)$ if $v \in \{m_k\}$. Otherwise, if $v \notin \{m_k\}$, then there are the following two cases:

1) When $k = 2$, v cannot be connected to any monitors, when both of m_1 and m_2 are removed. Therefore, $\{m_1, m_2\}$ is a 2-vertex-cut that separates $\mathcal{T}_0$ from $\mathcal{G}_{new}$, i.e., $\mathcal{T}_0$ does not include any nodes in $\mathcal{G}$ except for m_1 and m_2.

2) When $k \geq 3$, if any two nodes of $\mathcal{G}$ have been removed, v is connected to at least one real monitor, then v is always connected to the virtual monitors since there must be two virtual links between the real monitor and virtual monitors. Therefore, there does not exist any 2-vertex-cuts which can separate v from the virtual monitors in $\mathcal{G}_{new}$, i.e., v and the virtual monitors must be included in the same TC of $\mathcal{G}_{new}$. Hence, $v \in V(\mathcal{T}_0)$ since m'_1 and m'_2 are only included in $\mathcal{T}_0$.

Therefore, in both cases, if v is a real monitor or is connected to at least one real monitor when any two nodes of $\mathcal{G}$ have been removed, then $v \in V(\mathcal{T}_0)$. This completes the proof. ■

Lemma 7. *Any exterior path p cannot be identified with 2 monitors, one of which is the end node of p.*

Proof. In the proof of Theorem 10, we concluded that there are only 2 ways for a path p to achieve identifiability:

1. "On its own" as a virtual link, p is a cross-path or a shortcut.
2. By two identifiable segments that compose p. Note that if there are 3 or more identifiable segments such that $p = s_1 + s_2 + s_3$, we can always recursively reduce to the case of two segments by $p = s'_1 + s'_2$, $s'_1 = s_1 + s_2$ and $s'_2 = s_3$.

Then we prove the unidentifiability of an exterior path p from these 2 angle of views.

From the "virtual link" point of view. Firstly, it is obviously that p cannot be a cross-path because it is an exterior path, then p being a shortcut is the only way for it to be identifiable as a virtual link, which leads to an another path p^1 of known metric that shares the end nodes with p. Such p^1 is still an exterior path, it cannot be a cross-path either, then same as p, the identifiability of p^1 can only be achieved by

1. it is an another shortcut.

2. it has 2 identifiable segments that compose it.

If we still consider p^1 from the "virtual link" point of view, then p^1 must be a shortcut, leading to another exterior path p^2 that will fall into the same analysis procedure as p^1, i.e., p^2 is either another shortcut or it has 2 identifiable segments that compose it. If we continue to consider p^2 as a "virtual link", it must be another shortcut to be identifiable, which leads to another exterior path p^3. Recursively, if we consider all of these shortcuts coming along in the recursive process from"virtual link" point of view, then there will be an infinite series of exterior path $\{p^n\}$ that all are shortcuts. This is impossible by the fact that $\mathcal{G}$ is of finite structure. Therefore, either we can conclude that p is unidentifiable or there must be at least one exterior path p^k in $\{p^n\}$, whose identifiability is given by the the second way, i.e., it has 2 identifiable segments that compose it.

We next prove that case 2) is also impossible by showing that one of the two segments of p must be unidentifiable. For any combination of 2 segments that compose p, there must be one that is incident to the monitor, denoted as S_m (unless S_m is a link, S_m is still an exterior path), We claim that every S_m is unidentifiable, causing p to be unidentifiable.

And the proof that S_m is unidentifiable is similar to the proof that S_p is unidentifiable in Theorem 10. In conclusion, p is unidentifiable. ■

3.6.2 Main Proofs

Proof of Theorem 1

Any link in $\widetilde{\mathcal{T}}$ that incident to the vantage is a link that is crossing only one of the vantages, according to Lemma 4, such link is unidentifiable.

In the first place, $\mathcal{T}$ is 3-vertex-connected guarantees that $\widetilde{\mathcal{T}}$ is not a circle.

If $\widetilde{\mathcal{T}}$ is 3-vertex-connected, we have the following two conditions:

(a) $\mathcal{T}$ is 3-vertex-connected.

(b) $\mathcal{T} - l$ is 2-edge-connected for every interior link l of $\mathcal{T}$.

And to get measurement information for $\mathcal{T}$, any measurement path must go in $\mathcal{T}$ through one of its 2 vantages and go out from $\mathcal{T}$ through the other vantage, thus if only consider the identifiability of $\mathcal{T}$, the 2 vantages of $\mathcal{T}$ act like the monitor for every link in $\mathcal{T}$ (In case where the 2 vantages are 2 monitors, the statement is obviously consistent). Then by *Theorem.III.2* in [31], the above two conditions guarantee that the interior graph of $\mathcal{T}$, $\mathcal{H}(\mathcal{T})$, which is also the interior graph of $\widetilde{\mathcal{T}}$, is identifiable.

If $\widetilde{\mathcal{T}}$ is not 3-vertex-connected, since $\mathcal{T}$ is 3-vertex-connected for sure, then the degradation of connectivity of $\widetilde{\mathcal{T}}$ is caused by deleting the direct link l_{μ_1,μ_2}(if exist). After l_{μ_1,μ_2} is removed, there arises 2-vertex-cut or 2-bridge-cut in $\widetilde{\mathcal{T}}$, as illustrated in Fig. 3.29 and Fig. 3.30.

1. When there are only 2-vertex-cut in $\widetilde{\mathcal{T}}$, then $\widetilde{\mathcal{T}}$ is not a TC, we can continue partitioning TCs in $\widetilde{\mathcal{T}}$. And the continuing partitioned TCs in $\widetilde{\mathcal{T}}$ can only have 3 vantages or 4 vantages, e.g., in Fig. 3.29, $\mathcal{T}_1'$ (w.r.t. 3 vantages $\{\mu_1, v_1, v_2\}$), $\mathcal{T}_2'$ (w.r.t. 4 vantages, $\{v_1, v_2, v_3, v_4\}$) and $\mathcal{T}_3'$ (w.r.t. 3 vantages, $\{\mu_2, v_3, v_4\}$) are the continuing partitioned TCs of $\widetilde{\mathcal{T}}$. Links in TC that has 3 or 4 vantages are all identifiable according to *Proof of Theorem III.11* in [33].

2. When there are 2-bridge-cuts in $\widetilde{\mathcal{T}}$, the unidentifiability of 2-bridge-cut has been proved in [31]. Other links are in the continuing partitioned 3-vertex-connected TCs, which all has 3 vantages or 4 vantages, e.g., in Fig. 3.29, $\mathcal{T}_1'$ (w.r.t. 3 vantages $\{\mu_1, v_1, v_2\}$), $\mathcal{T}_2'$ (w.r.t. 4 vantages, $\{v_3, v_4, v_5, v_6\}$) and $\mathcal{T}_3'$(w.r.t. 3 vantages, $\{\mu_2, v_7, v_8\}$) are the continuing partitioned TCs of $\widetilde{\mathcal{T}}$. Links in 3-vertex-connected TC that has 3 or 4 vantages are all identifiable according to *Proof of Theorem III.11* in [33].

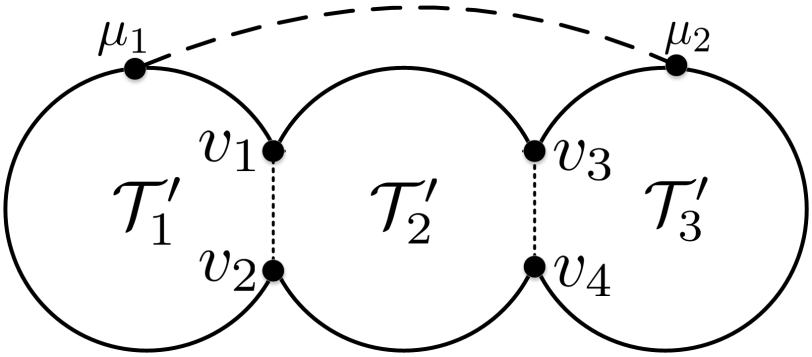


Figure 3.29: There are 2-vertex-cuts in $\widetilde{\mathcal{T}}$

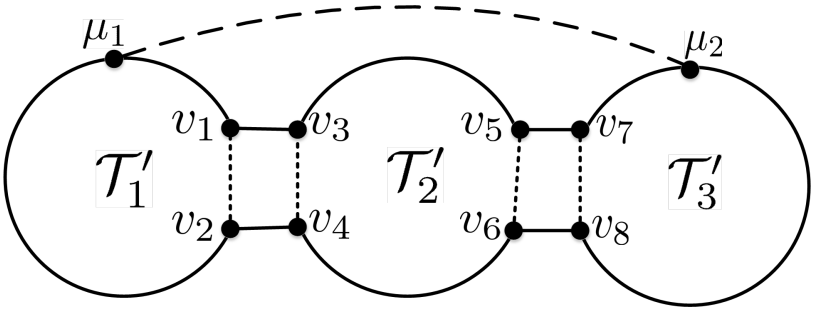


Figure 3.30: There are 2-bridge-cuts in $\widetilde{\mathcal{T}}$

Proof of Theorem 2

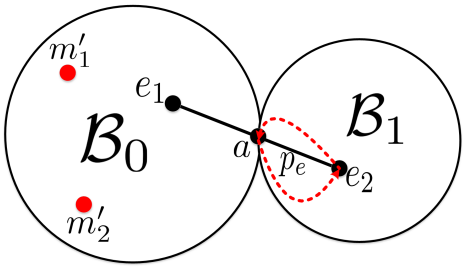


Figure 3.31: A path p belongs to category 1.

Category 1: In step 1, we have extended the original graph $\mathcal{G}$ to an extended graph $\mathcal{G}_{new}$, now we make argument with $\mathcal{G}_{new}$ herein. In $\mathcal{G}_{new}$, there are only two monitors, m_1' and m_2'. For a path p, if it has a vertex v satisfying that $v \in V(p)$ but $v \notin V(\mathcal{B}_0)$, then to measure the segment of p outside $\mathcal{B}_0$, denoted as p_e, any measurement path must go out $\mathcal{B}_0$ and then go back to $\mathcal{B}_0$ through the only agent of $\mathcal{B}_0$'s neighbor BC $\mathcal{B}_1$, as shown in Fig. 3.31. These measurement paths cannot avoid a loop, which is forbidden by the assumption, i.e., "MP is cycle-free". Hence, there is no MP that could contain p_e or any path outside $\mathcal{B}_0$, even if the segment of p within $\mathcal{B}_0$ is identifiable, p cannot be identified since the lack of knowledge of p_e's metric.

Category 2: In the first place, we denote the TC that contains the end nodes e_1 or e_2 of p as $\mathcal{T}_{e_1}$ and $\mathcal{T}_{e_2}$, and name the segment of p in $\mathcal{T}_{e_1}$ or $\mathcal{T}_{e_2}$ "tail of p". For example, in Fig. 3.32 (a), $\mathcal{T}_{e_1}$ is $\mathcal{T}_2^1$, $\mathcal{T}_{e_2}$ is $\mathcal{T}_2^2$, and the "tail of p" is $e_1\mu_3$ and $e_2\mu_6$. In the proof of Theorem 10 (refer to Section 3.6.1, we have concluded that there are only 2 ways for a path p to achieve identifiability:

1) *"On its own" as a virtual link, p is a cross-path or a shortcut.*

This cannot hold for the p that belongs to category 2, because for the paths whose end nodes are not in the same TC, it cannot be a cross-path or a shortcut.

2) *By two identifiable segments that compose p.* (Note that if there are 3 or more identifiable segments such that $p = s_1 + s_2 + s_3$, we can always recursively reduce to the case of two segments by $p = s_1' + s_2'$, $s_1' = s_1 + s_2$ and $s_2' = s_3$.)

We next prove that case 2) is also impossible by showing that one of the two segments of p must be unidentifiable.

From Lemma 4, we know that the "tail of p" is unidentifiable (it is only occupying one vantage in a 2-vantage TC), then in order to utilize the second identifiable condition, the choice of 2 segments of p cannot be "tail of p" + " the remained part, i.e., $p -$ tail". Thus, to divide p into 2 potential identifiable segments, the cut should be taken on the inner vertex of the "tail of p" or on the inner vertex of "$p-$tail". As an example shown in Fig. 3.32 (b), the whole graph $\mathcal{G}_{new}$ could be partitioned into 2 sub-graphs by the 2 vantages μ_3 and μ_4, the cut to divide p into 2 segments either is taken in $\mathcal{G}_1$ or $\mathcal{G}_2$ (except the vantage μ_3 since $e_1\mu_3$ is "tail of p", which is unidentifiable).

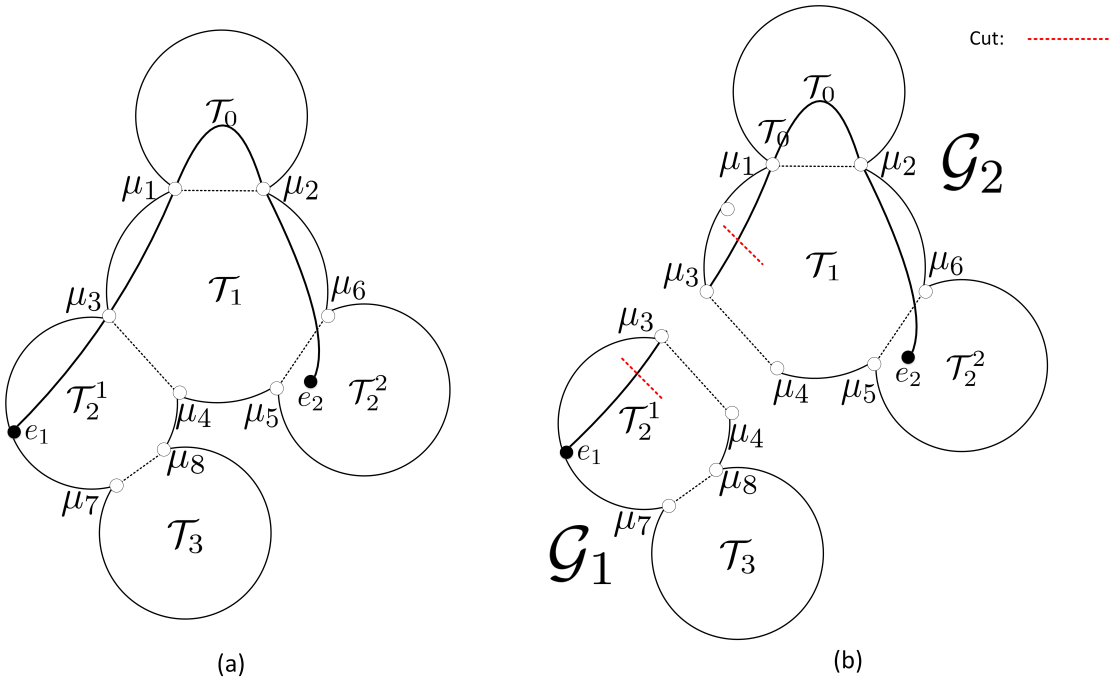


Figure 3.32: A path p belongs to category 2.

(a) *If the cut is taken in* $\mathcal{G}_1$, then in any combination of 2 segments, one is contained in $\mathcal{G}_1$, denoted as s_1, and the other one is crossing $\mathcal{G}_1$ and $\mathcal{G}_2$, denoted as s_c. Next we show that every s_c is unidentifiable.

The proof that s_c *is unidentifiable is as follows:*

Firstly, s_c cannot be identified as a virtual link because s_c cannot be a cross-path or a shortcut for its two end nodes are not in the same TC.

Secondly, the identifiability of s_c cannot be achieved by 2 identifiable segments that compose it. Division of s_c leads us to a new segment s_c^1 which is still crossing $\mathcal{G}_1$ and $\mathcal{G}_2$, and s_c^1 will go through the same analysis and then brings up a s_c^2. Recursively, we will end up reaching a segment s_c^n that is composed by one link in $\mathcal{G}_1$ and one link in $\mathcal{G}_2$. This s_c^n is unidentifiable for sure, because i), it cannot be a cross-path or a shortcut; ii), any combination of 2 segments of it are 2 unidentifiable segments. Thus, every s_c^i with $i \leq n$ and s_c are unidentifiable since each s_c^i being identifiable relies on s_c^{i+1} being identifiable. Therefore, s_c^1 being unidentifiable causes s_c to be unidentifiable.

(b) *If the cut is taken in* $\mathcal{G}_2$, then the divided 2 segments are one in $\mathcal{G}_2$, denoted as s_2 and the other one is crossing $\mathcal{G}_1$ and $\mathcal{G}_2$, denoted as s_c. Similar to the argument in a), we have that any s_c, i.e., the segment that is crossing $\mathcal{G}_1$ and $\mathcal{G}_2$, is unidentifiable. Then no matter what identifiability of the other segment s_2 is, p is unidentifiable because s_c is unidentifiable.

Proof of Lemma 1

In the very first place, we show that if a is a *shortcut* (refer to Section 3.6.1) of b (reversely, b is a *shortcut* of a), then the identifiability of a and b are equivalent: if a is identifiable, then b is identifiable by the definition of shortcut; if a is unidentifiable, we can only deduce b is unidentifiable since once b is identifiable, then a becomes identifiable because it is b's shortcut. Next we prove this lemma.

(1) If there is no real link between 2 vantages of a TC, a virtual link will be added in the process of partition $\mathcal{B}_0$ to TCs, then l_{μ_1,μ_2} is always considered as a link in the graph. And l_{μ_1,μ_2} is the shortcut of p, then the identifiability of l_{μ_1,μ_2} is equivalent to p.

(2) We need to prove that the sufficient part "l_{μ_1,μ_2} *and* p_e^s *are identifiable* $\Rightarrow$ *U-shape is identifiable*" and the necessary part "*U-shape is identifiable* $\Rightarrow$ l_{μ_1,μ_2} *and* p_e^s *are identifiable*" (For this necessary part, we prove its equivalent converse-negative condition "l_{μ_1,μ_2} *or* p_e^s *is unidentifiable* $\Rightarrow$ *U-shape is unidentifiable*"). And we will prove the sufficient and necessary part under different situations, which stems from the following analysis:

In the proof of Theorem 10 (refer to Section 3.6.1, we have concluded the only 2 ways for a path p to achieve identifiability as follows:

I. *"On its own" as a virtual link, p is a cross-path or a shortcut.*

It is obviously to see that a U-shape path cannot be identified as a virtual link because it cannot be a *cross-path* or a *shortcut*.

II. *By two identifiable segments that compose p.* (Note that if there are 3 or more identifiable segments such that $p = s_1 + s_2 + s_3$, we can always recursively reduce to the case of two segments by $p = s_1' + s_2'$, $s_1' = s_1 + s_2$ and $s_2' = s_3$.)

When considering its segments, note that no matter where to cut a U-shape p into 2 segments, one of the 2 segments from cutting is always an another U-shape piece, this is because l_{μ_1,μ_2} is a link, which does not have inner vertex, then l_{μ_1,μ_2} will always be in one of the segments, making this segment still a U-shape path. Recursively, such U-shape path also cannot be identified as a virtual link and we need to investigate its segments, by going through the same analysis and cutting, it will end up with a situation where no more U-shape to be cut, and there are l_{μ_1,μ_2} and other piece of segments of $p_{\mu_1 e_1}$ and $p_{\mu_2 e_2}$. Therefore, if p can become identifiable through its identifiable segments, the only way is by segments l_{μ_1,μ_2}, $p_{\mu_1 e_1}$ and $p_{\mu_2 e_2}$.

To study the identifiability of $\{p_{\mu_1 e_1}, p_{\mu_2 e_2}\}$, p_e^s plays an important role. Denote the TC with respect to the 2 vantages $\{\mu_1, \mu_2\}$ as $\mathcal{T}$, note that in the U-shape p, $p_{\mu_1 e_1}$ and $p_{\mu_2 e_2}$ in $\mathcal{T}$ are both exterior paths, which are both unidentifiable according to Lemma 7, but $p_{\mu_1 e_1} + p_{\mu_2 e_2}$ has a chance to become identifiable together, which relies on the identifiability of p_e^s: as argued in Lemma 4), the 2 vantages of $\mathcal{T}$ (μ_1 and μ_2) act as 2 monitor under the strongest assumption (every link is identifiable outside $\mathcal{T}$), under such condition, for $p_{\mu_1 e_1} + p_{\mu_2 e_2}$ to be identifiable,

it can **only** rely on the connecting paths between e_1 and e_2 to merge them together in a path from μ_1 and μ_2. p_e^s is the shortest one of all the connecting paths between e_1 and e_2, on the one hand, if p_e^s is identifiable (i.e., the value of its metric $W_{p_e^s}$ could be returned), by subtracting $W_{p_e^s}$ from the measurement value of $p_{\mu_1 e_1 e_2 \mu_2}$, we are able to obtain $W_{p_{\mu_1 e_1}+p_{\mu_2 e_2}}$; On the other hand, if p_e^s is unidentifiable, it must contain unidentifiable links, to recall Theorem 1, p_e^s must contain 2-bridge-cut links (only unidentifiable links inside a TC), because of the special topological status of 2-bridge-cut links, if the shortest path between e_1 and e_2 contains 2-bridge-cut link, any other path between e_1 and e_2 cannot avoid 2-bridge-cut link, thus we cannot find another identifiable p_e that helps identify $p_{\mu_1 e_1 e_2 \mu_2}$.

Namely, $p_{\mu_1 e_1} + p_{\mu_2 e_2}$ can only rely on p_e^s to become identifiable even though p is under the strongest assumption. It also can only rely on p_e^s to become identifiable in any weaker situations.

Based on the above analysis, the identifiability of U-shape p relies on p_e^s and l_{μ_1,μ_2}. Next we categorize the topological relation between U-shape p and p_e^s, and there are 2 cases that may occur:

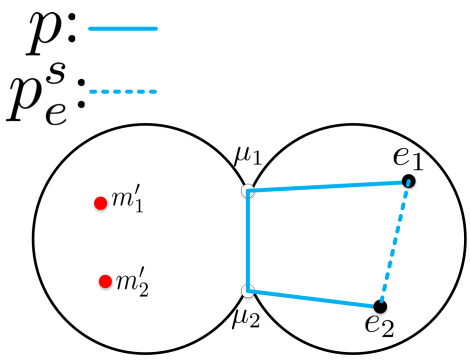


Figure 3.33: A U-shape that has disjoint shortest path between its end nodes.

(a) p_e^s ***is inner-vertex disjoint to*** p, i.e., other than the two end nodes e_1 and e_2, there is no joint vertex of p_e^s and p, as shown in Fig. 3.33.

In such case, $p_{\mu_1 e_1} + p_e^s + p_{\mu_2 e_2}$ becomes a D-shape, which is identifiable if and only if l_{μ_1,μ_2} is identifiable as we proved in *(1)*. Now we prove the sufficient and necessary part for this case.

Sufficient part.

If l_{μ_1,μ_2} is identifiable and p_e^s is identifiable (i.e., each link of p_e^s is identifiable by Theorem 3), then the D-shape $p_{\mu_1 e_1} + p_e^s + p_{\mu_2 e_2}$ is identifiable by *1)*, and the U-shape p is identifiable as

$$W_p = W_{p_{\mu_1 e_1}+p_e^s+p_{\mu_2 e_2}} - W_{p_e^s} + W_{l_{\mu_1,\mu_2}}$$

Necessary part.

If l_{μ_1,μ_2} (or p_e^s) is unidentifiable, then the D-shape $p_{\mu_1 e_1}+p_e^s+p_{\mu_2 e_2}$ is unidentifiable, destroying the only compensating way for $\{p_{\mu_1 e_1}, p_{\mu_2 e_2}\}$ to be identifiable. With the fact that every combination of 2 segments of U-shape p always contains one segment that is a U-shape piece,

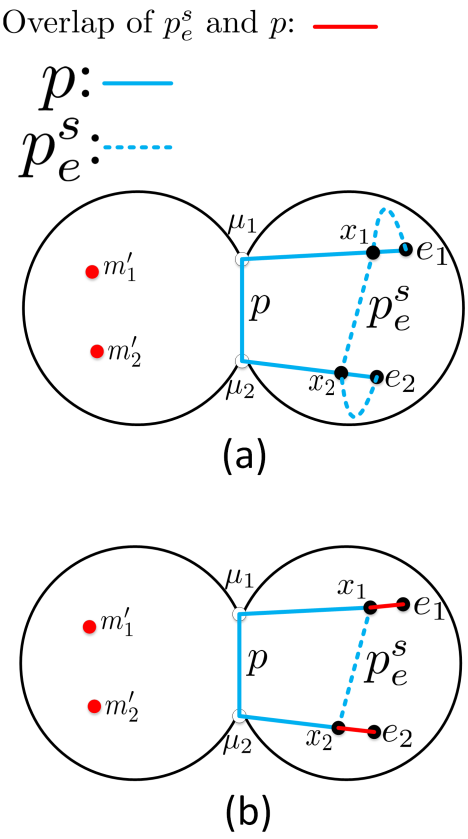


Figure 3.34: p_e^s and p has joint inner vertex.

such uncompensated U-shape is always unidentifiable, causing p to be unidentifiable.

(b) p_e^s ***and*** p ***is not inner-vertex disjoint***, i.e., other than the two end nodes e_1 and e_2, there exists joint vertex of p_e^s and p.

In such case, p_e^s cannot avoid intersecting at least one of $\{p_{\mu_1 e_1}, p_{\mu_2 e_2}\}$. Without loss of generality, we assume p_e^s intersects both $p_{\mu_1 e_1}$ and $p_{\mu_2 e_2}$, as illustrated in Fig. 3.34.

Regrading whether or not p_e^s is edge-disjoint to p, there are two possible sub-cases:

(i) p_e^s *is edge-disjoint to* p*, thus does not contain any part of* p, as shown in Fig. 3.34 (a).
In such case, $\mathcal{T}$ cannot be a circle but a 3-vertex-connected component. If x_1e_1 and x_2e_2 are removed, then the remained part is U-shape $p_{x_1\mu_1\mu_2x_2}$, denoted as p_u. p_u is able to have a disjoint path between x_1 and x_2. Then we prove the sufficient and necessary part for this case.

Sufficient part.

If l_{μ_1,μ_2} is identifiable and p_e^s are identifiable (i.e., each link of p_e^s is identifiable), then we have that x_1x_2 is identifiable, as in case 1), the U-shape p_u is identifiable. In p, we are left with x_1e_1 and x_2e_2 to consider, and we claim that each links of them is identifiable. Indeed, take x_1e_1 for example, if it contains unidentifiable links, which can only be 2-bridge-cut by Theorem 3, then $p_{\mu_1 e_1}$ must go through 2 links from the same 2-bridge-cut,

as shown in Fig. 3.35 (b), otherwise, if it only goes through one link that belongs to a 2-bridge-cut, as shown in Fig. 3.35 (c), the shortest path p_e^s in such case will never cross the 2-bridge-cut(if p_e^s goes through the 2-bridge-cut, it will not be the shortest path between e_1 and e_2), *resulting in no 2-bridge-cut can appear in* x_1e_1 *or* x_2e_2. Thus, if x_1e_1 contains 2-bridge-cut, p_e^s must contain 2-bridge-cut because a 2-bridge-cut is the only connecting way from e_1 to e_2. This contradicts the assumption that each link of p_e^s is identifiable.

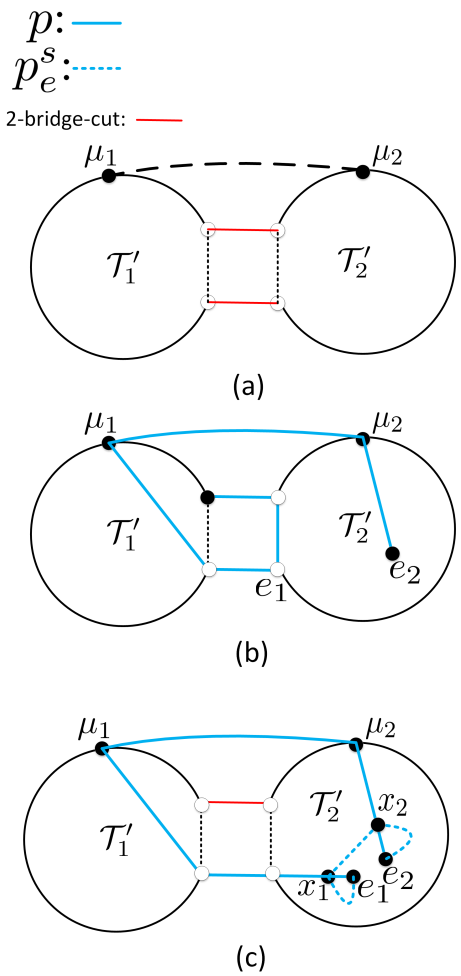


Figure 3.35: Cases where p contains 2-bridge-cut link.

Necessary part.

Same as the argument in *necessary part* for case (a).

(ii) p_e^s contains x_1e_1 and x_2e_2, as shown in Fig. 3.34 (b).
Sufficient part.

If l_{μ_1,μ_2} is identifiable and p_e^s are identifiable (i.e., each link of p_e^s is identifiable). We could have that the U-shape path $p_{\mu_1x_1x_2\mu_2}$ is identifiable from *(1)* in this proof. Besides, $x_1e_1 = p^s_{x_1e_1}$, $x_2e_2 = p^s_{x_2e_2}$, which are both identifiable, then we can obtain that the whole U-shape p is identifiable.

Necessary part.

Same as the argument in *necessary part* for case (a).

Proof of Theorem 3

(1) $\mathcal{T}$ is a circle.

p is in this TC $\mathcal{T}$, i.e., $\{\mu_1, \mu_2\} \not\subseteq V(p)$. p will fail the only 2 ways to become identifiable (argued in Theorem 10, refer to Section 3.6.1): i), every vertex in a circle is of 2 degree, which guarantees that every link or path in a circle cannot be a *cross-link/cross-path* or a *shortcut*. ii), every link in $\widetilde{\mathcal{T}}$ is unidentifiable, then any combination of 2 segments that compose p is 2 unidentifiable segments. Hence, p is unidentifiable.

(2) $\mathcal{T}$ is 3-vertex-connected.

Sufficient Part.

If every link in $L(p)$ is identifiable, i.e., every W_{l_i} is known. By the fact that p is composed by $L(p)$, then p is identifiable as the sum of the every $W_{l_i}\cdot$.

Necessary Part.

It is equivalent to prove its converse-negative condition, i.e., "if there is at least one unidentifiable link in p, then p is unidentifiable". From Theorem 1 we know that the unidentifiable links in $\mathcal{T}$ can only be i), exterior link; ii), 2-bridge-cut. So, if p contains an unidentifiable link, either it is an exterior link or a link from a 2-bridge-cut. And we prove the converse-negative condition by 2 steps: i), once p contains at least one exterior link in $\mathcal{T}$, p is unidentifiable. ii), p has no exterior link, but p contains at least a link from 2-bridge-cut, p is still unidentifiable.

i) p contains at least one exterior link in $\mathcal{T}$.

For a path p belongs to *single-TC* p, it could at most contain 4 exterior links, otherwise p forms a loop, violating the premise that "there is no loop in interested path".

i-1) If p only contains 1 exterior link, then p must be an exterior path, which has been proved unidentifiable in Lemma 7.

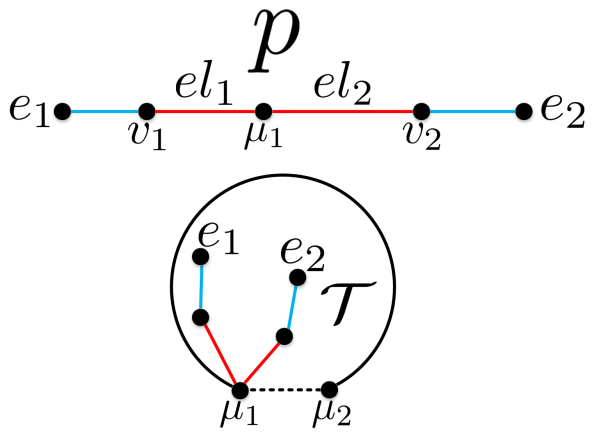


Figure 3.36: p contains 2 exterior links

i-2) If p contains 2 exterior links, el_1 and el_2, these 2 exterior links should be incident to the same vantage, as shown in Fig. 3.36, otherwise p becomes a D-shape. In such case, p is unidentifiable according to Lemma 4.

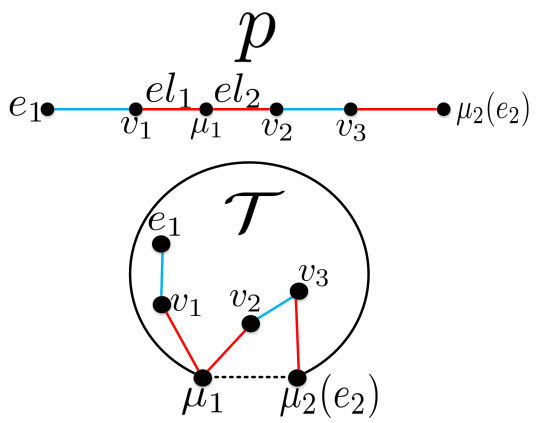


Figure 3.37: p contains 3 exterior links

i-3) If p contains 3 exterior links, el_1, el_2 and el_3, 2 of these 3 exterior links should be incident to the same vantage, and the third link is incident to the other vantage, as shown in Fig. 3.37.

In this case, p cannot be *cross-path* or a *shortcut* because it occupies the 2 vantages of $\mathcal{T}$, then p can only become identifiable through its segments. el_1, el_2 and el_3 are 3 links, which do not have inner vertex, thus to divide p into 2 segments, the cut can only be taken in e_1v_1 or v_3v_2.

If the cut is taken in any vertex in v_3v_2, the 2 resulting segments, s_1 and s_2, obtained this cutting are both unidentifiable. Indeed, for s_1 and s_2, one belongs to case 2. and one belongs to case 1.

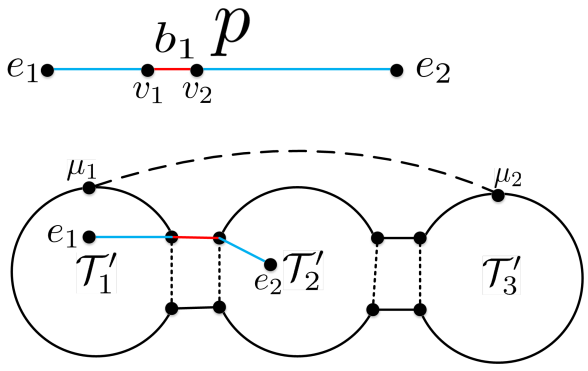


Figure 3.38: p has one link from 2-bridge-cut

If the cut is taken in e_1v_1, then for the 2 segments, s_3 and s_4, obtained by this cutting, one will still contain el_1, el_2 and el_3, and the other one is identifiable because it only contains identifiable links. Take s_3 as the one that contains el_1, el_2 and el_3, then to investigate such s_3, we will follow the same argument made before and come to the step of studying the segments of s_3.

Recursively, by the same process of analysis, eventually we can conclude that either there are 2 segments that compose p assuming "identifiable segment+unidentifiable segment" or any combination of 2 segments that compose p assumes "unidentifiable segment+unidentifiable segment", which both lead p to be unidentifiable.

i-4) If p contains 4 exterior links, the argument is similar to the case *(i-3)*.

ii) p has no exterior link, but p contains at least a link from 2-bridge-cut.

ii-1) *If p contains only* 1 *link from 2-bridge-cut*, denoted as b_1 in Fig. 3.38, then other links in p except b_1 are all identifiable. We can deduce that p is unidentifiable by proof of contradiction: if p is identifiable with metric W_p, W_p deducts other the metrics of other identifiable links in p is the metric of b_1, a contradiction.

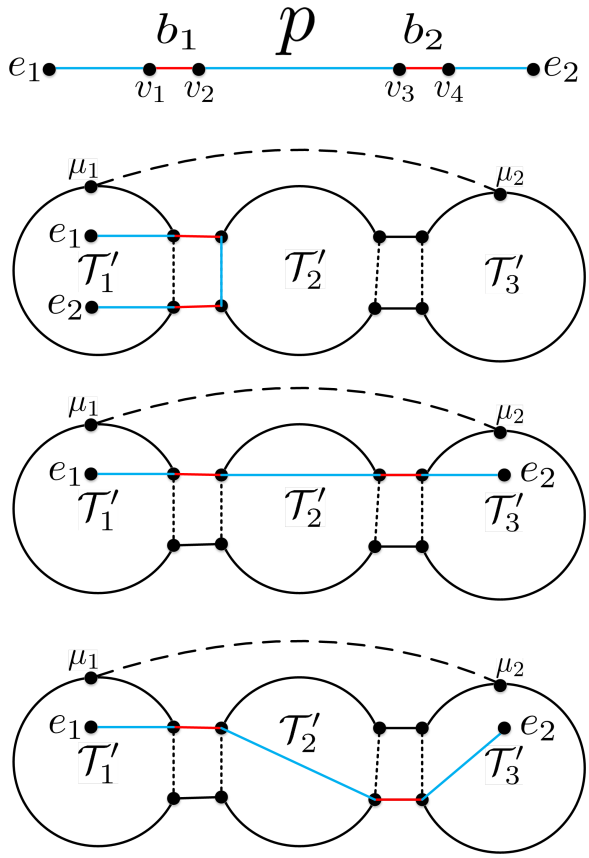


Figure 3.39: p has 2 links from 2-bridge-cut

ii-2) *If p contains* 2 *links from 2-bridge-cut*, denoted as b_1 and b_2, as shown in Fig. 3.39. If p is identifiable, then $b_1+v_2v_3+b_2$ is identifiable by $W_p - W_{e_1v_1} - W_{v_4e_2}$, but $b_1+v_2v_3+b_2$ cannot be an identifiable segment. Indeed, for the only two ways to identify a path (concluded in Theorem 10):

(i) $b_1 + v_2v_3 + b_2$ cannot be a *cross-path* or a *shortcut* no matter the b_1 and b_2 come from the same 2-bridge-cut or not, thus $b_1 + v_2v_3 + b_2$ cannot be identified as a virtual link.

(ii) $b_1 + v_2v_3 + b_2$ also cannot be deduced as identifiable through 2 identifiable segments that compose it. Since b_1 and b_2 are 2 links, which do not have any inner vertex, to divide p into 2 segments, s_1 and s_2, can only be taken in v_2v_3, e_1v_1 or e_2v_4.

If the cut is taken at any vertex in v_2v_3, then s_1 and s_2 both belongs to case *i-1*), which concludes the unidentifiability of s_1 and s_2.

If the cut is taken at any vertex in e_1v_1 *or* e_2v_4, it follows the similar recursive argument as case *i-3*) to conclude that s_1 and s_2 are both composed by an unidentifiable link (i.e., b_1 or b_2) and other identifiable part, then they are both unidentifiable (proof of contradiction as before).

(iii) *If* p *has* $k(k \geq 3)$ *links from 2-vertex-cut*, the argument is similar to the case where p has 2 links from 2-vertex-cut.

ii-3) *If* p *has* $k(k \geq 3)$ *links from 2-bridge-cut*, the argument is similar to the case where p has 2 links from 2-bridge-cut.

Proof of Theorem 4

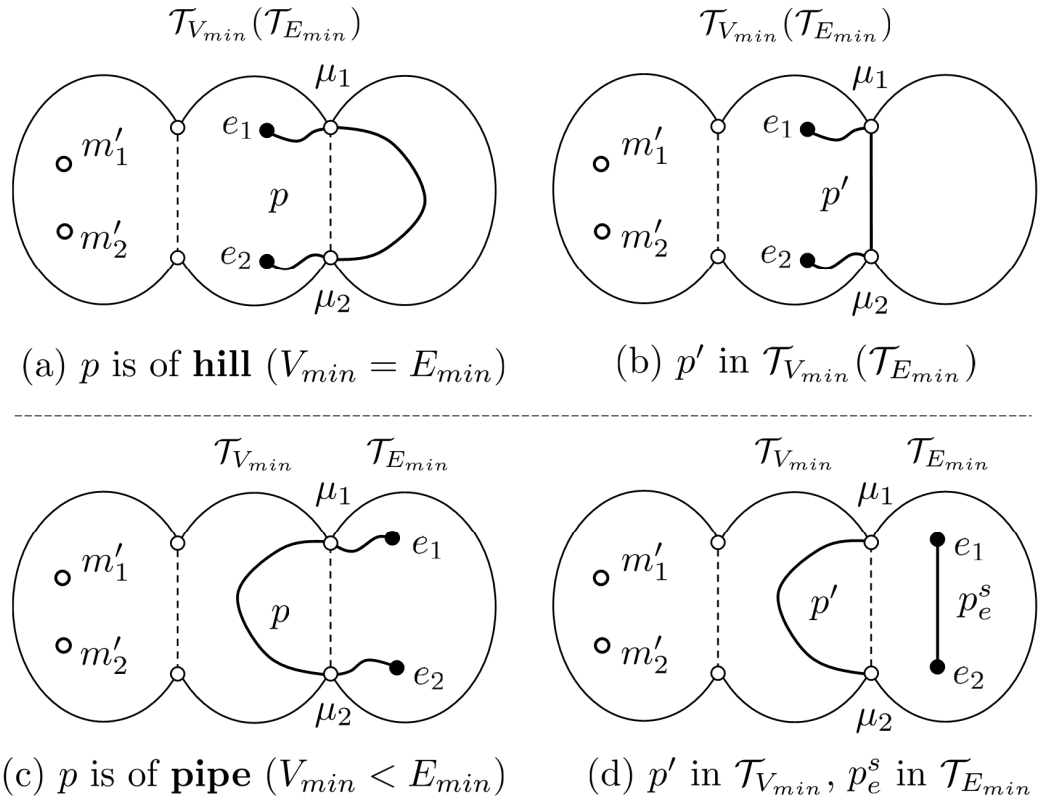


Figure 3.40: Two types of p in *category 3-case 2*: (a) hill, (b) transform of hill, (c) pipe, (d) transform of pipe.

- For a hill path p, as shown in Fig. 3.40 (a), p could be divided into 2 parts by the 2-vertex-cut $\{\mu_1, \mu_2\}$, one part is in $\mathcal{T}_{V_{min}}$ and the other part is the path $p_{\mu_1\mu_2}$ outside $\mathcal{T}_{V_{min}}$. According

to Lemma 1, the identifiability of $p_{\mu_1\mu_2}$ is same as the identifiability of l_{μ_1,μ_2}. Replacing $p_{\mu_1\mu_2}$ (composes p) by l_{μ_1,μ_2} (composes p'), the identifiability of p is same as p'.

- For a pipe path p, as shown in Fig. 3.40 (b), it could be seen as a "general" U-shape path where the link between μ_1 and μ_2 (l_{μ_1,μ_2}) is replaced by p'. Following the same argument to Lemma 1, such "general" U-shape path is identifiable if and only if p' and p_e^s are identifiable.

Proof of Theorem 5

Necessary part.

Given a graph $\mathcal{G}$ containing at least two monitors, and an unidentifiable path p, we prove that if p can be identified, then extra monitors are assigned so that the end nodes of p are included in $\mathcal{T}_0$. We prove this part by contradiction. Suppose on the contrary that p can be identified, but there is at least one end node of p that are not included in $\mathcal{T}_0$ after assigning extra monitors. Let e_1 and e_2 denote the end nodes of p. Based on the types of p, there are three cases as follows:

1) If p satisfies that its end nodes are in different BCs (a case listed in **category 1**), when there are extra monitors assigned, p is identifiable, and at least one end nodes of p are not included in $\mathcal{T}_0$, then there are the following two cases:

(a) if e_1 or e_2 is not included in $\mathcal{B}_0$, then p still belongs to **category 1** that is unidentifiable. A contradiction exists;

(b) if both e_1 and e_2 are included in $\mathcal{B}_0$, but at least one is not included in $\mathcal{T}_0$, then p belongs to **category 2** that is unidentifiable, since e_1 and e_2 are in different TCs. A contradiction exists.

2) If p belongs to **category 2**, then e_1 and e_2 are in different TCs before assigning extra monitors. When there are extra monitors assigned, p is identifiable, and we assume e_1 is not included in $\mathcal{T}_0$. No matter whether or not e_2 is included in $\mathcal{T}_0$, e_1 and e_2 will still be included in different TCs, i.e., p still belongs to **category 2** that is not identifiable. A contradiction exists.

3) If p belongs to *Single-TC*, then e_1 and e_2 are in same TC $\mathcal{T}$ with two vantages μ_1 and μ_2 before assigning extra monitors. When there are extra monitors assigned, p is identifiable, and we assume e_1 and e_2 are not included in $\mathcal{T}_0$. Then p will still be unidentifiable since there is nothing changed in $\mathcal{T}$. A contradiction exists.

Thus, if p is identifiable, then both of the end nodes of p are included in $\mathcal{T}_0$ after assigning extra monitors.

Sufficient part.

We prove that when there are extra monitors assigned and both of the end nodes of p are included in $\mathcal{T}_0$, then p is identifiable.

If both of the end nodes of p are included in $\mathcal{T}_0$, then p must belong to *single-TC* in $\mathcal{T}_0$, or belong to **hill** since $\mathcal{T}_0$ is the root TC (i.e., with the minimum number) in the tree structure of $\mathcal{B}_0$. If p belongs to **hill**, then p can be converted to a path p', which belongs to *single-TC* in $\mathcal{T}_{V_{min}}$, i.e., $\mathcal{T}_0$. Thus, according to Theorem 5, p is identifiable.

Therefore, if p is not identifiable and satisfies that (i) its end nodes are in different BCs (a case listed in **category 1**), or (ii) it belongs to **category 2**, or (iii) it belongs to *single-TC*, then p can

be identified if and only if extra monitors assigned so that both of the end nodes of p are included in $\mathcal{T}_0$. This completes the proof.

Proof of Lemma 2

Necessary part.

We prove that if $v \in V(\mathcal{H}(\mathcal{T}))$ is included in $\mathcal{T}_0$, then there is an extra monitor m assigned and satisfying one of the following conditions: (i) $m = v$, or (ii) $m \notin V(\mathcal{T})$, but assigned in a subgraph of $\mathcal{G}_{new}$, which does not included $\mathcal{T}$ and is obtained from $\mathcal{G}_{new}$ by removing v and its neighboring node in $\mathcal{T}$.

Let μ_1 and μ_2 denote the vantages of $\mathcal{T}$. We prove this part by contradiction. Suppose on the contrary that $v \in V(\mathcal{T}_0)$ after assigning extra monitors, but none of them satisfies the conditions (i) or (ii). Therefore, the extra monitors must be assigned in $\mathcal{T}$ except for v, or the *ancestor* TCs of $\mathcal{T}$ in the TC-tree of $\mathcal{B}_0$. Let v_1 and v_2 denote the neighboring nodes of v in $\mathcal{T}$. Then, any path that connects v to an existing monitor must include v_1 or v_2. Therefore, v cannot be connected to any monitors by removing v_1 and v_2, and v is not a monitor, then $v \notin V(\mathcal{T}_0)$ according to Lemma 6. A contradiction exists.

Hence, if a node $v \in V(\mathcal{H}(\mathcal{T}))$ is included in $\mathcal{T}_0$, then there must be an extra monitor assigned, which satisfies the conditions (i) or (ii).

Sufficient part.

We prove that if there is an extra monitor assigned and satisfying the condition (i) or (ii), then $v \in V(\mathcal{T}_0)$.

1) If there is an extra monitor m assigned to v, i.e., $m = v$, then we directly have $v \in V(\mathcal{T}_0)$ based on Lemma 6.

2) If there is an extra monitor m assigned that satisfies the condition (ii), then m has been assigned in the *descendant* TC of $\mathcal{T}$ in the TC-tree of $\mathcal{B}_0$. Then v can be connected to m without going through its neighboring node v_1 or v_2. And m can be connected to at least two given monitors through v_1 and v_2. Therefore, after removing any two nodes in $\mathcal{G}$, v must be connected to at least one monitor. Thus $v \in V(\mathcal{T}_0)$ based on Lemma 6.

For both cases, we have proved that if there is an extra monitor assigned and satisfying the condition (i) or (ii), then $v \in V(\mathcal{T}_0)$. This completes the proof.

Proof of Lemma 3

Necessary part.

We prove that if $v \in V(\mathcal{H}(\mathcal{T}))$ is included in $\mathcal{T}_0$, then there is an extra monitor m assigned and satisfying one of the following conditions: (i) $m = v$, or (ii) $m \notin V(\mathcal{T})$, but assigned in a subgraph of $\mathcal{G}_{new}$, which does not included $\mathcal{T}$ and is obtained from $\mathcal{G}_{new}$ by removing v and its neighboring node in $\mathcal{T}$.

Let μ_1 and μ_2 denote the vantages of $\mathcal{T}$. We prove this part by contradiction. Suppose on the contrary that $v \in V(\mathcal{T}_0)$ after assigning extra monitors, but none of them satisfies the conditions (i) or (ii). Therefore, the extra monitors must be assigned in $\mathcal{T}$ except for v, or the *ancestor* TCs of $\mathcal{T}$ in the TC-tree of $\mathcal{B}_0$. Let v_1 and v_2 denote the neighboring nodes of v in $\mathcal{T}$. Then, any path that

connects v to an existing monitor must include v_1 or v_2. Therefore, v cannot be connected to any monitors by removing v_1 and v_2, and v is not a monitor, then $v \notin V(\mathcal{T}_0)$ according to Lemma 6. A contradiction exists.

Hence, if a node $v \in V(\mathcal{H}(\mathcal{T}))$ is included in $\mathcal{T}_0$, then there must be an extra monitor assigned, which satisfies the conditions (i) or (ii).

Sufficient part.

We prove that if there is an extra monitor assigned and satisfying the condition (i) or (ii), then $v \in V(\mathcal{T}_0)$.

1) If there is an extra monitor m assigned to v, i.e., $m = v$, then we directly have $v \in V(\mathcal{T}_0)$ based on Lemma 6.

2) If there is an extra monitor m assigned that satisfies the condition (ii), then m has been assigned in the *descendant* TC of $\mathcal{T}$ in the TC-tree of $\mathcal{B}_0$. Then v can be connected to m without going through its neighboring node v_1 or v_2. And m can be connected to at least two given monitors through v_1 and v_2. Therefore, after removing any two nodes in $\mathcal{G}$, v must be connected to at least one monitor. Thus $v \in V(\mathcal{T}_0)$ based on Lemma 6.

For both cases, we have proved that if there is an extra monitor assigned and satisfying the condition (i) or (ii), then $v \in V(\mathcal{T}_0)$. This completes the proof.

Proof of Theorem 6

Proof of part (i).

We prove that *all paths in* $\mathbf{P}$ *can be identified.* If $\mathcal{G}_{new}$ is 2-vertex-connected, then $\mathcal{G}_{new}$ includes only one BC $\mathcal{B}_0$. And each path $p \in \mathbf{P}$ meets that $V(p) \subseteq V(\mathcal{B}_0)$, i.e., it belongs to **category 2** or **category 3**. If p belongs to **category 3**, then p can be converted to 1 or 2 paths, which belongs to *single-TC*. Therefore, only two types of paths need to be identified, one is **category 2** and the other one is *single-TC*.

If $\mathcal{B}_0$ includes only one TC $\mathcal{T}_0$, then each path $p \in \mathbf{P}$ satisfies $L(p) \subseteq L(\mathcal{H}(\mathcal{T}))$, since p does not include any virtual links, which are the exterior links of $\mathcal{T}_0$. Therefore, p is identifiable according to Lemma 5.

Otherwise, $\mathcal{B}_0$ includes more than one TCs, then there exists at least one Edge-TC. For a path $p \in \mathbf{P}$, there are the following two cases based on the types of p:

1) If p belongs to **category 2**, let e_1 and e_2 be the end nodes of p, then e_1 and e_2 are in different TCs. According to Theorem 5, p can be identifiable if the end nodes of p are included in $\mathcal{T}_0$. Therefore, we only focus on whether or not e_1 and e_2 are included in $\mathcal{T}_0$. For each end node e_i of p, there are the following two cases:

(a) $e_i \in V(\mathcal{H}(\mathcal{T}))$, where $\mathcal{T}$ is an Edge-TC;

(b) $e_i \in V(\mathcal{H}(\mathcal{T}))$, where $\mathcal{T}$ is not an Edge-TC

For case (a), no matter $\mathcal{T}$ is a circle or not, Algorithm 2 has to assign an extra monitor in $\mathcal{T}$ for e_i, since $|E(p) \cap V\mathcal{T}| = 1$, i.e., e_i. Thus, $e_i \in V(\mathcal{T}_0)$.

For case (b), there must be at least one *descendant* TC that has been assigned monitor, since each of the remaining Edge-TCs has been assigned at least one extra monitors by Algorithm 2. If

$\mathcal{T}$ is 3-vertex-connected, then $e_i \in V(\mathcal{T}_0)$ based on Lemma 3. Otherwise, $\mathcal{T}$ is a circle. In such case, Algorithm 1 has to be executed according to the conditions in Lemma 2, and then $e_i \in V(\mathcal{T}_0)$.

Therefore, if p belongs to **category 2**, then the end nodes of p must be included in $\mathcal{T}_0$ after executing Algorithm 2, and then p is identifiable.

2) If p belongs to *single-TC*, then there is a TC $\mathcal{T}$ satisfying that $E(p) \cap V(\mathcal{H}(\mathcal{T})) \neq \emptyset$, and there are the following two cases based on types of $\mathcal{T}$:

(a) $\mathcal{T}$ is a circle;

(b) $\mathcal{T}$ is not a circle but an Edge-TC;

(c) $\mathcal{T}$ is neither a circle nor an Edge-TC

For case (a), let e_1 and e_2 denote the end nodes of p. If e_i satisfies $d(e_i) = 2$, then Algorithm 2 calls Algorithm 1 to place extra monitors, which can make e_i meet $e_i \in V(\mathcal{T}_0)$. Otherwise, $d(e_i) \geq 2$, which means there must be at least one descendant TC of $\mathcal{T}$, which is an Edge-TC that has been assigned extra monitors. Thus, according to Lemma 2, $e_i \in V(\mathcal{T}_0)$ when $d(e_i) \geq 2$. Therefore, p is identifiable since $E(p) \subseteq V(\mathcal{T}_0)$.

For case (b), if $L(p)$ is identifiable, then p is identifiable, otherwise, p satisfies that (i) $|E(p) \cap V(\mathcal{H}(\mathcal{T}))| = 1$ (i.e., p includes an exterior link), or (ii) $|E(p) \cap V(\mathcal{H}(\mathcal{T}))| = 2$ and p includes unidentifiable links. Thus, Algorithm 2 assigns an extra monitor in $\mathcal{H}(\mathcal{T})$, and then $V(p) \subseteq V(\mathcal{T}_0)$ according to Lemma 3. Hence, p is identifiable since $E(p) \subseteq V(\mathcal{T}_0)$.

For case (c), there must be at least one descendant TC of $\mathcal{T}$, which is an Edge-TC that has been assigned extra monitors. According to Lemma 3, $V(p) \subseteq V(\mathcal{T}_0)$. Thus, p is identifiable since $E(p) \subseteq V(\mathcal{T}_0)$.

For all case above, each $p \in \mathbf{P}$ is identifiable.

Proof of part (ii).

We prove that no placement can identify all the interested paths with a smaller number of extra monitors. We assume that $M^*(\mathcal{G}, \mathbf{P})$ is the optimal monitor placement for $\mathbf{P}$ in $\mathcal{G}$.

Algorithm 2 places monitors in two steps: first, it places monitors for Edge-TCs until each Edge-TC includes at least one monitor (*lines 7-26*); Second, it places monitors for remaining circular TCs, which are not processed in the first step (*lines 27-30*). Therefore, there are the following three cases of TC $\mathcal{T}$, which may be assigned extra monitors by Algorithm 2:

(a) $\mathcal{T}$ is an Edge-TC and a circle;

(b) $\mathcal{T}$ is an Edge-TC but not a circle;

(c) $\mathcal{T}$ is a circle but not an Edge-TC.

For *case (a) and (c)*, no matter $\mathcal{T}$ is an Edge-TC or not, if there is a node $v \in V(\mathcal{H}(\mathcal{T}))$ that satisfies $d(v) = 2$ and $v \in E(p)$, where $p \in \mathbf{P}$, then Algorithm 1 is called to place extra monitors for v. Let v_1 and v_2 denote its neighboring nodes in $\mathcal{T}$. Then the extra monitor m for v is assigned on one of the following three nodes: (i) the assistant node of l_{v,v_1}, or (ii) the assistant node of l_{v,v_2}, or (iii) node v. Suppose that $M^*(\mathcal{G}, \mathbf{P})$ does not include a node satisfying the conditions (i), (ii) or (iii),

then $v \notin V(\mathcal{T}_0)$ according to Lemma 2. And then p is not identifiable since $v \in E(p)$. Therefore, m must be included in $M^*(\mathcal{G}, \mathbf{P})$.

For *case (b)*, when an Edge-TC $\mathcal{T}$ is not a circle (i.e., 3-vertex-connected), if there is at least one path p that satisfies $E(p) \cap V(\mathcal{H}(\mathcal{T})) \neq \emptyset$, then Algorithm 2 may assign an extra monitor in $\mathcal{H}(\mathcal{T})$. There are the following two cases:

(b)-1 $|E(p) \cap V(\mathcal{H}(\mathcal{T}))| = 1$;

(b)-2 $|E(p) \cap V(\mathcal{H}(\mathcal{T}))| = 2$

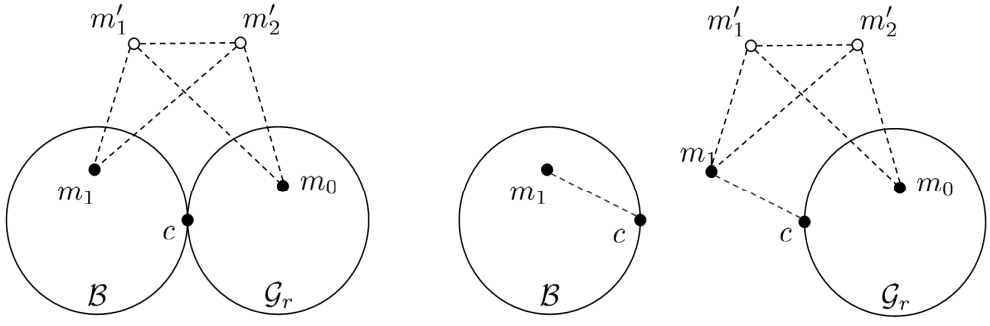
For *case (b)-1*, if p belongs to **category 2**, then p is not identifiable. Otherwise, p belongs to *single-TC* and is also unidentifiable according to Theorem 3, since it includes exterior links that are unidentifiable. Suppose that $M^*(\mathcal{G}, \mathbf{P})$ does not include a node in $\mathcal{H}(\mathcal{T})$, then p is unidentifiable, since at least one end node of p is not included in $\mathcal{T}_0$.

For *case (b)-2*, Algorithm 2 only assigns the extra monitor when p is unidentifiable, i.e., p includes unidentifiable links. Similar to case (a), we suppose that $M^*(\mathcal{G}, \mathbf{P})$ does not include a node in $\mathcal{H}(\mathcal{T})$, then p is unidentifiable according to Theorem 5, since at least one end node of p is not included in $\mathcal{T}_0$.

Therefore, for all the cases above, each assigned monitor is included in the optimal monitor placement $M^*(\mathcal{G}, \mathbf{P})$.

Proof of Theorem 7

Algorithm 3 first obtains a trimmed graph $\mathcal{G}^t$ by trimming a number of Edge-BCs, which do not include any links within interested paths. And based on Theorem V.7 in [20], $M^*(\mathcal{G}^t, \mathbf{P}) \subseteq M^*(\mathcal{G}, \mathbf{P})$. Therefore, each Edge-BC $\mathcal{B}$ includes at least one link within interested paths, and we prove that $M^*(\mathcal{B}, \mathbf{P}_\mathcal{B}) \subseteq M^*(\mathcal{G}^t, \mathbf{P})$.



(a) $\mathcal{G}_{new}$, $|M(\mathcal{B})| = 1$ (b) $\mathcal{B} + l_{m_1 c}$ is a TC within $\mathcal{G}_{new}$

Figure 3.41: A Edge-BC with k monitors (I).

For an Edge-BC $\mathcal{B}$, let $c_\mathcal{B}$ denote the cut-point in $\mathcal{B}$, a path $p \in \mathbf{P}_\mathcal{B}$ satisfies that one of the following conditions:

(a) $p \in \mathbf{P}$, and $V(p) \subseteq V(\mathcal{B})$;

(b) p is a sub-path of p' between v and $c_\mathcal{B}$, where $p' \in \mathbf{P}$ belongs to *multi-BC* and $v = E(p') \cap V(\mathcal{B})$.

$M^*(\mathcal{G}, \mathbf{P})$ must include at least one monitor in $\mathcal{B}$. Otherwise, $\mathcal{B}$ can be separated from $\mathcal{G}^t_{new}$ by removing $c_{\mathcal{B}}$, i.e., $\mathcal{B}$ is not included in $\mathcal{B}_0$. And then the paths in $\mathbf{P}_{\mathcal{B}}$ are not identifiable since they belong to **category 1**. Therefore, $|M^*(\mathcal{B}, \mathbf{P}_{\mathcal{B}})| \geq 1$.

Algorithm 3 selects an initial monitor m_1 from $\mathcal{B}$ by enumeration, and $\{m_1, c_{\mathcal{B}}\}$ is a 2-vertex-cut of $\mathcal{G}^t_{new}$ (e.g., Fig. 3.41). And then Algorithm 2 is called to identify all paths in $\mathbf{P}_{\mathcal{B}}$ with a minimum number of extra monitors in $\mathcal{B}$ except for m_1 and $c_{\mathcal{B}}$.

When $|M^*(\mathcal{B}, \mathbf{P}_{\mathcal{B}})| \geq 2$, we prove that $M^*(\mathcal{B}, \mathbf{P}_{\mathcal{B}}) \subseteq M^*(\mathcal{G}^t, \mathbf{P})$ by contradiction. Suppose on contrary that a monitor m is included in $\mathcal{B}$ but is not included in $M^*(\mathcal{G}^t, \mathbf{P})$, and $\mathbf{P}_{\mathcal{B}}$ is identifiable. If m is removed, then there must be at least one node $v \notin \mathcal{T}_0$, which is an end node of $p \in \mathbf{P}_{\mathcal{B}}$. Suppose v is included in a TC $\mathcal{T}$ within $\mathcal{B}$. According to Lemma 2 and Lemma 3, $v \in \mathcal{T}_0$ needs an extra monitor assigned in the descendants of $\mathcal{T}$, which must be included in $\mathcal{B}$. Therefore, when m is removed, no matter how many monitors are assigned in $\mathcal{G}_r$, p is not identifiable. A contradiction exists.

Therefore, $M^*(\mathcal{B}, \mathbf{P}_{\mathcal{B}}) \subseteq M^*(\mathcal{G}^t, \mathbf{P}) \subseteq M^*(\mathcal{G}, \mathbf{P})$. This completes this proof.

3.6.3 Proof of Theorem 8

Algorithm 3 first obtains a trimmed graph $\mathcal{G}^t$ by trimming a number of Edge-BCs, which do not include any links within interested paths. And based on Theorem V.7 in [20], an optimal monitor placement for $\mathcal{G}^t$ is also an optimal monitor placement for $\mathcal{G}$. Therefore, we focus on to prove that the monitor placement in $\mathcal{G}^t$ is optimal.

Proof of part (i).

We prove that *all paths in* $\mathbf{P}$ *can be identified*. Depending on the number of BCs within $\mathcal{G}^t$, there are two cases as follows:

1) $\mathcal{G}^t$ includes only one BC

Since $\mathcal{G}^t$ is 2-vertex-connected and Algorithm 3 selects two initial monitors by exhaustive enumeration first, $\mathcal{G}^t_{new}$ can be constructed and must be 2-vertex-connected. Therefore, Algorithm 2 is called to place monitors for identifying all paths in $\mathbf{P}$. And then, according to Theorem 6, all paths in $\mathbf{P}$ can be identified.

2) $\mathcal{G}^t$ includes more than one BC

In this case, Algorithm 3 first assigns monitors for Edge-BCs until each Edge-BC includes at least one monitor. And then Algorithm 3 calls Algorithm 2 to place monitors for interested paths, and all paths can be identified since $\mathcal{G}^t_{new}$ is 2-vertex-connected.

Proof of part (ii).

1) $\mathcal{G}^t$ includes only one BC

In this case, exhaustive enumeration is used for selecting a pair of initial monitors to execute Algorithm 2. And for each pair of initial monitors, Algorithm 2 can generate a monitor placement with the minimum number of extra monitors based on Theorem 6. Therefore, the optimal monitor placement must be the one with the minimum number of monitors.

2) $\mathcal{G}^t$ includes more than one BC

In this case, Algorithm 3 first assigns monitors for Edge-BCs until each Edge-BC includes at least one monitor. And for each Edge-BC $\mathcal{B}_i$, there is $M^*(\mathcal{B}_i, \mathbf{P}_{\mathcal{B}_i}) \subseteq M^*(\mathcal{G}^t, \mathbf{P})$ based on Theorem 7.

When each Edge-BC $\mathcal{B}_i$ includes at least one monitor, $\mathcal{G}^t_{new}$ can be constructed with the monitors $\cup M^*(\mathcal{B}_i, \mathbf{P}_{\mathcal{B}_i})$, which must be 2-vertex-connected. Thus, Algorithm 2 is called to identify all paths in $\mathbf{P}$ with the minimum number of extra monitors. Therefore, $M^*(\mathcal{G}^t, \mathbf{P})$ is optimal.

Hence, for both cases, we have prove that $M^*(\mathcal{G}^t, \mathbf{P})$ is optimal, and then $M^*(\mathcal{G}, \mathbf{P})$ is optimal.

Chapter 4

Controlling the Total Error Bound in Bound-based Network Tomography

4.1 Overview

In this chapter, we extend Boolean-based network tomography to bound-based network tomography where the lower and upper bounds are derived for unidentifiable links. we develop different solutions for i) obtaining the tightest total error bound, and ii) deploying a new monitor such that the total error bound could be maximally reduced.

4.2 Boolean-based Tomography to Bound-based Tomography

Existing solutions for network performance tomography are mostly *Boolean based*: they determine whether or not a given set of links/paths are identifiable[1] (i.e., Boolean), and if yes, the inferred values on the link/paths of interest are returned. If a link/path is not identifiable, no useful information about the performance on that link/path is provided in Boolean-based network tomography.

Given a set of measurements, a link/path may not be identifiable. However, the performance upper bound and lower bound on that link/path can be derived, which are also important to the ISPs. After all, what an ISP cares is whether the service-level agreement (SLA) with the customers is violated or not, i.e., whether the performance metric is within the performance bound specified in the SLA.

We thus shift our focus towards performance bounds: what are the tightest upper and lower bounds of the performance metric on the links/paths of interest? We call this shift of focus as *bound-based* network performance tomography.

[1] Identifiable means the value on a link/path can be uniquely determined.

Clearly, *bound-based* network tomography is more general and extends *Boolean-based* network tomography, because in *bound-based* network tomography the upper and lower bounds are the same for identifiable links/paths. Whenever a link/path is not identifiable, bound-based network tomography tells the upper and lower values on the link/path. In practice, bound-based network tomography offers more information to an ISP to diagnose performance problems and manage its network resources with less measurement overhead.

We study bound-based network tomography with *additive* (e.g., delay) metrics for links, where the metric of an end-to-end path is the sum of metric of all the links on the path. Bound-based network tomography, while extending Boolean-based network tomography only slightly in concept, drastically changes the landscape of solution space as illustrated in the following example.

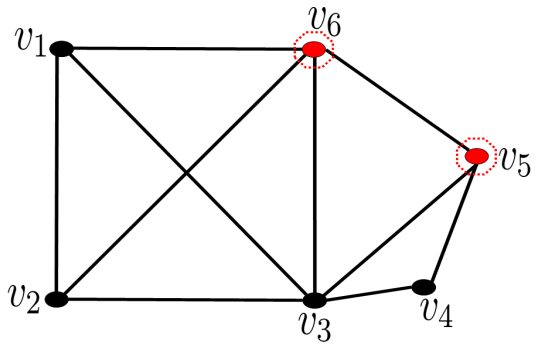


Figure 4.1: Initial monitor deployment.

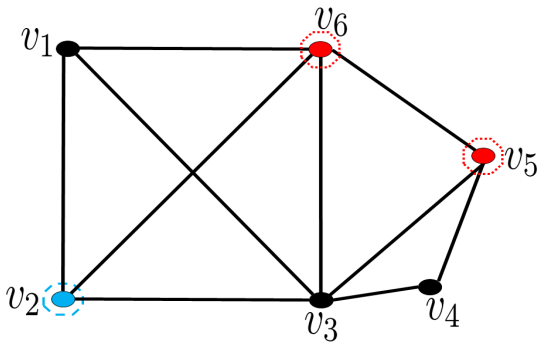


Figure 4.2: New monitor deployment with Boolean-based network tomography.

Motivating Example: Fig. 4.1 shows an example network with 6 nodes and 10 links, and 2 initial monitors are placed at v_5 and v_6, respectively. Assume that the link metrics are $\{x_{1,2} = 7, x_{2,3} = 3, x_{3,4} = 10, x_{4,5} = 13, x_{5,6} = 8, x_{1,6} = 1, x_{1,3} = 5, x_{2,6} = 4, x_{3,6} = 6, x_{3,5} = 3\}$, which are unknown in advance and should be inferred through end-to-end measurements with monitors.

With the Boolean-based network tomography introduced in [31], only links $l_{5,6}$ and $l_{1,2}$ are identifiable. The metrics on other links remain unknown. In contrast, we can derive the bounds on unidentifiable links, using the bound-based network tomography disclosed later in this chapter, as

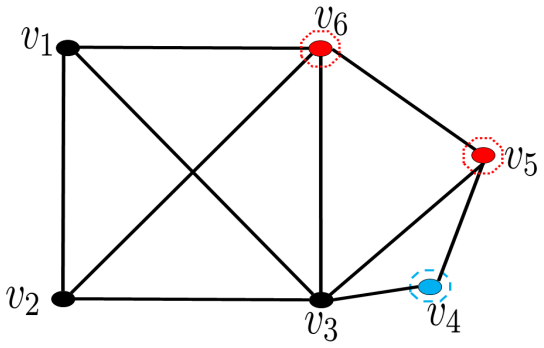


Figure 4.3: New monitor deployment with bound-based network tomography.

$\mathcal{B}(x_{3,5}) = [0, 7]$, $\mathcal{B}(x_{3,4}) = [0, 27]$, $\mathcal{B}(x_{2,6}) = [3, 10]$, $\mathcal{B}(x_{3,6}) = [2, 9]$, $\mathcal{B}(x_{1,3}) = [2, 9]$, $\mathcal{B}(x_{4,5}) = [0, 27]$, $\mathcal{B}(x_{2,3}) = [0, 7]$, $\mathcal{B}(x_{1,6}) = [0, 7]$. If we denote the *error bound* of a link metric as the inferred upper bound minus the inferred lower bound, the *total error bound* of all links is 96.

After we obtain the tightest lower and upper bounds on unidentifiable links, a natural question is: how to further tighten the total error bound by deploying an extra monitor? Boolean-based network tomography cannot answer this question directly. One may wonder if the monitor placement solution to minimize the total number of unidentifiable links [31] would also lead to the tightest total error bound. This solution is shown in Fig 4.2 where the new monitor is placed at v_2 (in blue color). The total number of unidentifiable links is reduced to 2, which are $l_{3,4}$ and $l_{4,5}$ with new bounds $\mathcal{B}(x_{3,4}) = [0, 23]$ and $\mathcal{B}(x_{4,5}) = [0, 23]$, respectively. The total error bound is reduced to 46. With the bound-based network tomography which will be presented in this chapter, we should deploy the new monitor at v_4, as shown in Fig 4.3, which reduces the total error bound to 16 even if the total number of unidentifiable links is 4 (i.e., $\mathcal{B}(x_{1,6}) = [0, 4]$, $\mathcal{B}(x_{2,3}) = [0, 4]$, $\mathcal{B}(x_{1,3}) = [2, 6]$, and $\mathcal{B}(x_{2,6}) = [3, 7]$). Clearly, bound-based network tomography finds a better solution w.r.t. reducing the total error bound.

Motivated by the above example, we aim to answer the following two questions in this chapter:

- Given a set of deployed monitors, *how to infer the tightest lower and upper bounds of unidentifiable links?*
- Given a set of deployed monitors, *how to deploy extra monitors to minimize the total error bound?*

4.3 Derive the Tightest Total Error Bound

End-to-end delays along MPs form a linear system. We use the example in Fig. 4.4 to illustrate the concept. The network has two monitors in red. First, the end-to-end delays along all MPs between the two monitors form the following linear system, where $x_{i,j}$ denotes the delay on link $l_{i,j}$ and w_l denotes the end-to-end delay on MP l.

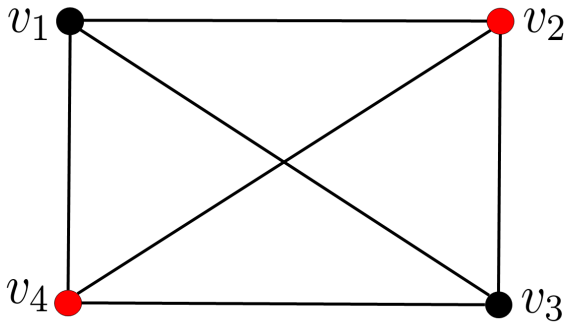


Figure 4.4: An example network.

$$\begin{cases} x_{1,2} + x_{1,4} = w_1 \\ x_{2,4} = w_2 \\ x_{2,3} + x_{3,4} = w_3 \\ x_{1,2} + x_{1,3} + x_{3,4} = w_4 \\ x_{2,3} + x_{1,3} + x_{1,4} = w_5 \end{cases} \tag{4.1}$$

The above linear system could be written into $R'\mathbf{x} = \mathbf{w}'$, where

$$R' = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix} \tag{4.2}$$

$$\mathbf{x} = \begin{pmatrix} x_{1,2} & x_{1,3} & x_{1,4} & x_{2,3} & x_{2,4} & x_{3,4} \end{pmatrix}^\intercal \tag{4.3}$$

$$\mathbf{w}' = \begin{pmatrix} w_1 & w_2 & w_3 & w_4 & w_5 \end{pmatrix}^\intercal \tag{4.4}$$

We also call R' the *measurement matrix* corresponding to $\mathbf{x}$. From linear system (4.1), we can derive that $x_{2,4} = w_2$ and $x_{1,3} = (w_4 + w_5 - w_3 - w_1)/2$. Since we can derive the values of $x_{2,4}$ and $x_{1,3}$, we could move the known values to the right-hand-side of (4.1) to form a new linear system:

$$\begin{cases} x_{1,2} + x_{1,4} = w_1 \\ 0 = w_2 - x_{2,4} \\ x_{2,3} + x_{3,4} = w_3 \\ x_{1,2} + x_{3,4} = w_4 - x_{1,3} \\ x_{2,3} + x_{1,4} = w_5 - x_{1,3} \end{cases} \tag{4.5}$$

Since $x_{2,4}$ only appears in the second equation, it cannot be used to determine the bounds for other unidentifiable links. As such, the second equation $0 = w_2 - x_{2,4}$ is out of our interest and can be safely removed. The resulting linear system is the final mathematical model useful for our purpose.

$$\begin{cases} x_{1,2} + x_{1,4} = w_1 \\ x_{2,3} + x_{3,4} = w_3 \\ x_{1,2} + x_{3,4} = w_4 - x_{1,3} \\ x_{2,3} + x_{1,4} = w_5 - x_{1,3} \end{cases} \tag{4.6}$$

which could be simplified as $R\mathbf{x} = \mathbf{w}$ where

$$R = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix} \tag{4.7}$$

$$\mathbf{x} = \begin{pmatrix} x_{1,2} & x_{1,3} & x_{1,4} & x_{2,3} & x_{2,4} & x_{3,4} \end{pmatrix}^{\intercal} \tag{4.8}$$

$$\mathbf{w} = \begin{pmatrix} w_1 & w_3 & w'_4 & w'_5 \end{pmatrix}^{\intercal} \tag{4.9}$$

where $w'_4 = w_4 - x_{1,3}$ and $w'_5 = w_5 - x_{1,3}$.

Remark 2. *In the rest of this paper, the linear system model (LSM), $R\mathbf{x} = \mathbf{w}$, by default refers to the final linear system after the above preliminary processing. We stress that the LSM is non-invertible, because links in the LSM are unidentifiable (otherwise there is no need for bound-based tomography.).*

Since *LSM* $R\mathbf{x} = \mathbf{w}$ is non-invertable, the rank of R (i.e., the number of linearly independent measurement paths) is less than the number of links in $\mathcal{G}$. In linear algebra [10], the solutions of non-invertable *LSM* form a solution space, which is constructed by free variables. In other words, the solution $\mathbf{x}$ is presented in the form that every other pivot variable (i.e., non-free variable) is the linear combination of free variables.

4.3.1 Utilization of Free Variables

To derive the tightest total bound of undetermined variables in the *LSM*, we first introduce the following property of free variables [10].

Proposition 1. *For a non-invertible linear system $R\mathbf{x} = \mathbf{w}$ having n variables, there are different combinations of n' variables that can serve as free variables, where $n' = n - rank(R)$ [10].*

As an example, the solution space of linear system (4.6) can be represented as

$$\begin{cases} x_{1,2} = w_1 - w'_5 - x_{2,3} \\ x_{1,4} = w'_5 - x_{2,3} \\ x_{3,4} = w_3 - x_{2,3} \end{cases} \tag{4.10}$$

if we select $x_{2,3}$ as the free variable, or

Algorithm 4 Calculate the Tightest $\mathcal{TEB}$

Input: non-invertible *LSM*
Output: the tightest $\mathcal{TEB}$

1: Obtain the *TNB* of every variable in the *LSM*
2: Sort the variables in descending order by the upper bound in *TNB*, denoted as $\mathcal{O} = \{x_1^s, x_2^s, \cdots, x_n^s\}$
3: Re-arrange the LSM (by switching columns of the LSM matrix) to obtain the measurement matrix corresponding to the re-ordered variables
4: Find the reduced row echelon form (RREF) of the measurement matrix, based on which the set of free variables, denoted by F_c, is determined
5: For each free variable x_f in F_c, sum over $|TNB(x_f)|$ to S_f
6: For each pivot variable x_p, find the bound interval $B_c(x_p)$ based on free variable in F_c; sum over $|TNB(x_p) \cap B_c(x_p)|$ to S_p
7: **return** $\mathcal{TEB} = S_f + S_p$

$$\begin{cases} x_{1,4} = w_1 - x_{1,2} \\ x_{2,3} = w_5' - w_1 + x_{1,2} \\ x_{3,4} = w_1 + w_3 - w_5' - x_{1,2} \end{cases} \tag{4.11}$$

if we select $x_{1,2}$ as the free variable.

While the same solution space can be represented in different forms, we need to determine which representation leads to the tightest total error bound. This question is answered in the next section.

4.3.2 Algorithm to Obtain the Tightest Total Error Bound

Let $\mathcal{TEB} = \sum \mathcal{B}(x_i)$ denote the total error bound of an *LSM*, where $\mathcal{B}(x_i)$ is the length of bound interval for x_i. We first define the concept of *natural bound interval*:

Definition 8. ***Natural bound interval (NBI):*** *In an LSM, the natural bound interval of a variable x_i is the interval $[0, b_i]$, where b_i is the (end-to-end) measurement value corresponding to an equation that contains x_i. If there are multiple equations containing the variable x_i, the* ***tightest natural bound interval (TNB)*** *of x_i, denoted by* $\mathrm{TNB}(x_i)$*, is the interval $[0, b_{min}]$ where b_{min} is the smallest value among all the (end-to-end) measurements corresponding to those equations.*

The steps to obtain the tightest $\mathcal{TEB}$ are presented in Algorithm 4. While the pseudo-code is self-explaining, we note that in Step 3, once the measurement matrix is given (according to the re-ordering variables), the RREF is uniquely determined [10], based on which the pivot/free variables can be determined. In Step 4, a reduced row echelon form (RREF) can be easily found with Gauss-Jordan elimination [10]. In Step 5, for each free variable x_f in F_c, its final bound interval is its *TNB*. Its error bound is thus $|TNB(x_f)|$, where $|.|$ denotes the length of the interval. In Step 6, since each pivot variable x_p is a linear combination of the variables in F_c, we can obtain a bound interval for x_p, denoted as $B_c(x_p)$, by plugging free variables' *TNB*s into the linear combination. The intersection of $B_c(x_p)$ and $TNB(x_p)$, i.e., $TNB(x_p) \cap B_c(x_p)$, is the final bound interval of x_p.

Remark 3. *Algorithm 4 improves the algorithm proposed in [19]. The worst-case time complexity in [19] is super-polynomial of n where n is the number of unidentifiable links, but the worst-case*

time complexity of Algorithm 4 in this paper is polynomial of n, i.e., the complexity for sorting n values and the complexity of calculating RREF [10]. The former can be done in $O(n \log n)$; and the latter can be done in $O(mn^2)$ where m is the number of equations (i.e., measurement paths). From Remark 12, we can practically build the least measurement paths that are needed to derive the tightest $\mathcal{TEB}$ in polynomial time.

4.3.3 Theoretical Guarantee

Theorem 11. *The total error bound $\mathcal{TEB}$ obtained in Algorithm 4 is the minimum $\mathcal{TEB}$ that could be derived from the LSM.*

Proof of Theorem 11

We first prove the following lemma:

Lemma 8. *For a measurement matrix R (including all possible MPs), its reduced row-echelon form (RREF) is a matrix whose element can only be -1, 0, or 1.*

Proof. After the pre-processing introduced in Section II of the paper, we only need to consider a measurement matrix which contains *only* unidentifiable links. According to [46], there are only 3 types of unidentifiable links, i.e., *exterior links, 2-bridge-cuts (2-b-c), links in a circle TC*. Essentially, a link in a circle TC could be deemed as either an *exterior link* (incident to only one of the 2 vantages), or a *2-b-c* link.

Since we can always decompose a graph into TCs, we first consider the unidentifiable links and the measurement matrix associated with these links in a single TC. There are only two possible cases:

$$R = \begin{array}{c} \begin{array}{cccccccc} W_{a_1} & \cdots & W_{a_{k_1}} & W_{c_1} & \cdots & W_{c_{k_2}} \end{array} \\ \left(\begin{array}{cc|cccc} 1 & & 1 & & & \\ 1 & & & 1 & & \\ \vdots & & & & \ddots & \\ 1 & & & & & 1 \\ \hline & 1 & 1 & & & \\ & 1 & & 1 & & \\ & \vdots & & & \ddots & \\ & 1 & & & & 1 \\ \hline & \ddots & & & \vdots & \\ \hline & 1 & 1 & & & \\ & 1 & & 1 & & \\ & \vdots & & & \ddots & \\ & 1 & & & & 1 \end{array}\right) \end{array} \tag{4.12}$$

Case 1: When there are no *2-b-c* links in the TC (in this case *exterior links* must exist), let $\{a_1, a_2, \cdots, a_{k_1}\}$ and $\{c_1, c_2, \cdots, c_{k_2}\}$ denote the two groups of exterior links incident to the 2

distinct vantages (monitors), respectively. The measurement matrix is equivalent to the following matrix R (blank space means zero elements), because every interior link is identifiable and thus excluded from the matrix.

The RREF of the above matrix is

$$R_t = \left(\begin{array}{cccc|cccc} 1 & & & & & & & 1 \\ & 1 & & & & & & 1 \\ & & \ddots & & & & & 1 \\ & & & 1 & & & & 1 \\ \hline & & & & 1 & & & -1 \\ & & & & & 1 & & -1 \\ & & & & & & \ddots & -1 \\ & & & & & & & 1 \quad -1 \end{array}\right)$$

where the upper block contains k_1 rows, and the lower block contains $k_2 - 1$ rows.

Case 2: When there are *2-b-c* links (i.e., a pair of *2-b-c* links b_1 and b_2), the measurement matrix (denoted as R^1 to differentiate it from (4.12)) will be in the form shown in (4.13):

The RREF of R^1 is

$$R_t^1 = \left(\begin{array}{cccc|cccc} 1 & & & & & & & 1 \\ & 1 & & & & & & 1 \\ & & \ddots & & & & & \vdots \\ & & & 1 & & & & 1 \\ \hline & & & & 1 & & & -1 \\ & & & & & 1 & & -1 \\ & & & & & & \ddots & \vdots \\ & & & & & & 1 & -1 \\ \hline & & & & & & & 1 \quad -1 \end{array}\right)$$

where the upper block contains k_1 rows, the middle block contains $k_2 - 1$ rows, and the lower block contains 1 row.

Similarly, for the case where there are multiple *2-b-c* links, we just need to add two more columns, for each *2-b-c*, as the last two columns into the right end of matrix (4.13), which will also turn into an RREF with element $\{-1, 0, 1\}$.

The above argument shows that inside a TC, the measurement matrix would end up with an RREF that only has element $\{-1, 0, 1\}$. When the TC tree of network $\mathcal{G}$ has multiple TCs, as the analysis in Section IV of the paper, for a MP to reach TC $\mathcal{T}$, it needs to traverse the ancestors TCs of $\mathcal{T}$. If its ancestor $\mathcal{T}_a$ is a $\mathcal{T}_{3vc}$, then the only unidentifiable links this MP needs to traverse are 2 exterior links of this ancestor TC (each incident to one of $\mathcal{T}_a$'s vantage). If $\mathcal{T}_a$ is a $\mathcal{T}_{circle}$, then this MP will go through all the links in $\mathcal{T}_{circle}$ except the direct link l_{μ_3,μ_4} (where μ_3 and μ_4 are the vantages of $\mathcal{T}_a$'s child). Adding those unidentifiable links from ancestor TCs is essentially the

same as adding *2-b-c* links into the measurement matrix inside a TC, meaning that the RREF only contains $\{-1, 0, 1\}$.

When all TCs are considered together, the full measurement matrix is formed by merging all sub-matrices (corresponding to each TC) as independent blocks into a big matrix, whose RREF is the formed by merging all sub-matrices' RREFs together. Lemma 8 is thus proved.

■

$$R^1 = \begin{array}{c} \begin{array}{cccccccccc} W_{a_1} & \cdots & W_{a_{k_1}} & W_{c_1} & \cdots & W_{c_{k_2}} & W_{b_1} & W_{b_2} \end{array} \\ \left(\begin{array}{cccccc|cc} 1 & & & 1 & & & 1 & \\ 1 & & & & 1 & & 1 & \\ \vdots & & & & & \ddots & \vdots & \\ 1 & & & & & 1 & 1 & \\ \hline & 1 & & 1 & & & 1 & \\ & 1 & & & 1 & & 1 & \\ & \vdots & & & & \ddots & \vdots & \\ & 1 & & & & 1 & 1 & \\ \hline & & \ddots & & & \vdots & \vdots & \\ \hline & & 1 & 1 & & & 1 & \\ & & 1 & & 1 & & 1 & \\ & & \vdots & & & \ddots & \vdots & \\ & & 1 & & & 1 & 1 & \\ \hline 1 & & & 1 & & & & 1 \\ 1 & & & & 1 & & & 1 \\ \vdots & & & & & \ddots & & \vdots \\ 1 & & & & & 1 & & 1 \\ \hline & 1 & & 1 & & & & 1 \\ & 1 & & & 1 & & & 1 \\ & \vdots & & & & \ddots & & \vdots \\ & 1 & & & & 1 & & 1 \\ \hline & & \ddots & & & \vdots & & \vdots \\ \hline & & 1 & 1 & & & & 1 \\ & & 1 & & 1 & & & 1 \\ & & \vdots & & & \ddots & & \vdots \\ & & 1 & & & 1 & & 1 \end{array}\right) \end{array} \tag{4.13}$$

With Lemma 8, we next prove Theorem 11.

Proof. We first prove the case where any $n - r$ variables can serve as free variables. In this case, let's assume that the free variable combination providing tightest $\mathcal{TEB}$ is $\{x_{r+1}, x_{r+2}, \ldots, x_n\}$, then

the solution form should be arranged as

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_r \\ x_{r+1} \\ x_{r+2} \\ x_{r+3} \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_r \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \begin{pmatrix} c_{r+1,1} \\ c_{r+1,2} \\ \vdots \\ c_{r+1,r} \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} x_{r+1} + \begin{pmatrix} c_{r+2,1} \\ c_{r+2,2} \\ \vdots \\ c_{r+2,r} \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} x_{r+2} + \cdots + \begin{pmatrix} c_{n,1} \\ c_{n,2} \\ \vdots \\ c_{n,r} \\ 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix} x_n \tag{4.14}$$

where the first r variables are pivot variables. Moreover, $c_{i,j} \in \{-1, 0, 1\}$ ($i = r, r+1, \cdots, n$, $j = 1, 2, \cdots, n$) in (4.14) according to Lemma 8.

We will show that any other solution form, whose free variables' total $|TNB|$ is strictly greater than the total $|TNB|$ of $\{x_{r+1}, x_{r+2}, \ldots, x_n\}$, would lead to a larger total error bound. To change (4.14) into another solution form, at least one pivot variable should exchange with one free variable. Without loss of generality, we let pivot variable x_r exchange with free variable x_{r+1} which satisfies condition $|TNB(x_{r+1})| < |TNB(x_r)|$ (otherwise the total $|TNB|$ of new free variables is equal to the total $|TNB|$ of new free variables) and show that the resulting total error bound becomes larger.

Exchanging pivot variable x_r and free variable x_{r+1} could be done in the rth equation in (4.14)

$$x_r = d_r + c_{r+1,r}x_{r+1} + c_{r+2,r}x_{r+2} + \cdots + c_{n,r}x_n$$

where every $c_{i,r}$ ($i = r+1, r+2, \cdots, n$) is -1, 0 or 1 by Lemma 8. Under the given condition that any $n - r$ variables can be used as free variables, $c_{r+1,r}$ cannot be 0 (otherwise we cannot exchange x_{r+1} with x_r). Hence, we have

$$x_{r+1} = \begin{cases} d_r - x_r + \cdots + c_{n,r}x_n, & \text{when } c_{r+1,r} = -1 \\ -d_r + x_r - \cdots - c_{n,r}x_n & \text{when } c_{r+1,r} = 1 \end{cases} \tag{4.15}$$

Furthermore, Equation (4.15) can be written as

$$x_{r+1} = \begin{cases} -x_r + C & \text{when } c_{r+1,r} = -1 \\ x_r - C & \text{when } c_{r+1,r} = 1 \end{cases} \tag{4.16}$$

where $C = d_r + x_{r+2}c_{r+2,r} + \cdots + x_n c_{n,r}$.

From now on, we only illustrate on the case $x_{r+1} = -x_r + C$, since the other case $x_{r+1} = x_r - C$ follows the same analysis.

After the exchange, the new solution form could be split into 3 parts:

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{r-1} \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_{r-1} \end{pmatrix} + \begin{pmatrix} c_{r+1,1} \\ c_{r+1,2} \\ \vdots \\ c_{r+1,r} \end{pmatrix} (-x_r + C) + \begin{pmatrix} c_{r+2,1} \\ c_{r+2,2} \\ \vdots \\ c_{r+2,r} \end{pmatrix} x_{r+2} + \cdots + \begin{pmatrix} c_{n,1} \\ c_{n,2} \\ \vdots \\ c_{n,r} \end{pmatrix} x_n \tag{4.17}$$

$$\begin{cases} x_{r+1} = -x_r + C \\ x_r = x_r \end{cases} \tag{4.18}$$

$$\begin{pmatrix} x_{r+2} \\ x_{r+3} \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} x_{r+2} + \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix} x_{r+3} + \cdots + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix} x_n \tag{4.19}$$

Then we show that the above new form results in a larger $\mathcal{TEB}$ than that from the solution form (4.14). Observe that the affected equations by such an exchange are the first $r+1$ equations in (4.17) and the equations in (4.18), while the equations in (4.19) stay the same. Comparing from the bottom to the top, we have three observations:

1. Assume $TNB(x_r) = [0, b_r]$, $TNB(x_{r+1}) = [0, b_{r+1}]$. After the exchange, the total $|\,TNB|$ of free variable is increased by $\mathcal{I}_f = |\,TNB(x_r)| - |\,TNB(x_{r+1})| = b_r - b_{r+1}$.

2. Now we study the r-th equation $x_{r+1} = -x_r + C$ in the new solution form, which is equivalent to $x_r = -x_{r+1} + C$ in the original solution form. While mathematically the same, they do lead to different bound intervals: assume $C \in [c_1, c_2]$ ($c_1, c_2 \in \mathbf{R}$), consequently, $B_c(x_{r+1}) = [-b_r + c_1, c_2]$ and $B_c(x_r) = [-b_{r+1} + c_1, c_2]$, which leads to the bound intervals of x_r and x_{r+1}, respectively, as:

$$\mathcal{B}(x_{r+1}) = \begin{cases} [-b_r + c_1, c_2] \cap [0, b_{r+1}] & \text{if } c_2 > 0 \\ [0, b_{r+1}] & \text{if } c_2 \leq 0 \end{cases}$$

$$\mathcal{B}(x_r) = \begin{cases} [-b_{r+1} + c_1, c_2] \cap [0, b_r] & \text{if } c_2 > 0 \\ [0, b_r] & \text{if } c_2 \leq 0 \end{cases}$$

   Comparing the length of $\mathcal{B}(x_{r+1})$ and $\mathcal{B}(x_r)$, we have:

   - If $c_2 \leq 0$, then $|\mathcal{B}(x_{r+1})| - |\mathcal{B}(x_r)| = b_{r+1} - b_r < 0$, whose absolute value is equal to $\mathcal{I}_f$. In such case, the total bound interval for x_r and x_{r+1}, denoted as $\mathcal{TEB}(x_r + x_{r+1})$, is unaltered after the exchange.
   - If $c_2 > 0$, there are 3 sub cases:
     (a) If $0 < c_2 < b_{r+1}$, then $\mathcal{B}(x_{r+1}) = \mathcal{B}(x_r) = [0, c_2]$. Thus, $|\mathcal{B}(x_r)| + |\,TNB(x_{r+1})| = c_2 + b_{r+1} \leq |\mathcal{B}(x_{r+1})| + |\,TNB(x_r)| = c_2 + b_r$. $\mathcal{TEB}(x_r + x_{r+1})$ is increased after the

exchange.

(b) If $b_{r+1} \leq c_2 \leq b_r$, then $\mathcal{B}(x_{r+1}) = [0, c_2]$. There are 2 cases for the bound interval of $\mathcal{B}(x_r)$

- $\square$ $\mathcal{B}(x_r) = [c_1 - b_{r+1}, c_2]$ when $c_2 - c_1 + b_{r+1} < c_2$, and in this case, $|\mathcal{B}(x_r)| = c_2 - c_1 + b_{r+1}$, which is less than $|\mathcal{B}(x_{r+1})| = c_2$. With $|TNB(x_{r+1})| \leq |TNB(x_r)|$, we have $|\mathcal{B}(x_r)| + |TNB(x_{r+1})| \leq |\mathcal{B}(x_{r+1})| + |TNB(x_r)|$, thus $\mathcal{TEB}(x_r + x_{r+1})$ is increased after the exchange.
- $\square$ $\mathcal{B}(x_r) = [0, c_2]$ when $c_2 - c_1 + b_{r+1} \geq c_2$, and in this case, $|\mathcal{B}(x_r)| = |\mathcal{B}(x_{r+1})| = c_2$. With $|TNB(x_{r+1})| \leq |TNB(x_r)|$, we have $|\mathcal{B}(x_r)| + |TNB(x_{r+1})| \leq |\mathcal{B}(x_{r+1})| + |TNB(x_r)|$, thus $\mathcal{TEB}(x_r + x_{r+1})$ is also increased after the exchange.

(c) If $c_2 > b_r$, then $\mathcal{B}(x_{r+1}) = [c_1 - b_r, b_{r+1}]$, $\mathcal{B}(x_r) = [c_1 - b_{r+1}, b_r]$. Then, $|\mathcal{B}(x_{r+1})| = |\mathcal{B}(x_r)| = b_r + b_{r+1} - c_1$. With $|TNB(x_{r+1})| \leq |TNB(x_r)|$, we have $|\mathcal{B}(x_r)| + |TNB(x_{r+1})| \leq |\mathcal{B}(x_{r+1})| + |TNB(x_r)|$, thus $\mathcal{TEB}(x_r + x_{r+1})$ is increased after the exchange.

From the above analysis we can see $\mathcal{TEB}(x_r + x_{r+1})$ is either unaltered or increased, i.e., the total length of $\mathcal{B}(x_r)$ and $\mathcal{B}(x_{r+1})$ will stay the same or increase, after the exchange.

3. For any equation that belongs to the first $r-1$ equations, it could be written as

$$x_i = c_{r+1,i}(-x_r + C) + D \tag{4.20}$$

where $D = d_i + c_{r+2,i}x_{r+2} + \cdots + c_{n,i}x_n$ (for $i = 1, 2, \cdots, r-1$). Following the similar argument in 2), we could conclude that $|\mathcal{B}(x_i)|$ is also invariant or increased after the exchange.

Therefore, the total bound interval for all variable $\mathcal{TEB} = \sum \mathcal{B}(x_i)$ is non-decreasing after this exchange. Iteratively, if we continue to switch pivot variable with free variable, $\mathcal{TEB} = \sum \mathcal{B}(x_i)$ is non-decreasing. This concludes the proof for the case where any $n - r$ variables can serve as free variables.

For the case where not any $n - r$ variables can serve the free variables, Algorithm 1 in the paper selects the tightest free variable combination due to the ordering process in Step 2. We then can apply the same analysis as above but with a restriction that only eligible free variable combinations are considered after the exchange.

■

4.4 Deploying New Monitors to Reduce the Total Error Bound

So far we have the tightest $\mathcal{TEB}$ under the existing monitor deployment. A follow-up question is: if we are allowed to put an extra monitor, where should we put it so that $\mathcal{TEB}$ is maximally reduced by the new monitor? This question is unsolvable without extra assumptions, because we are not allowed to use a trial-and-error method. Hence, we use the mean value of the lower and upper bounds on each link to estimate the delay of a path between an existing monitor and a candidate node, should the new monitor be deployed at the candidate node. *The solution and theorem presented in this section are based on this assumption.*

To solve the above problem, we first show that the place of the new monitor could be determined at the TC level, and analyze how the TC tree structure evolves if we add a new monitor (Section 4.4.1). After illustrating the transform of TC tree, we then show the right place for new monitor deployment in the scenario of tandem TC tree (Section 4.4.2). Based on analysis for tandem TC tree, we finally develop the solution for a general TC tree (Section 4.4.3).

4.4.1 Total Error Bound Reduction Analysis

After partitioning G_{new} into TCs recorded in a TC tree, *the place of the new monitor, m_{new}, could be determined at the TC level.* This is based on two observations.

First, for a particular TC $\mathcal{T}$, placing m_{new} at different *interior nodes* of $\mathcal{T}$ (i.e., nodes in $\mathcal{T}$ except vantage nodes) will result in the same link identifiability [31,46], because i) if $\mathcal{T}$ is a 3-vertex-connected TC, then no matter where we put m_{new} inside $\mathcal{T}$, it will lead to the same new TC tree structure, based on which the link identifiability is determined, ii) if $\mathcal{T}$ is a circle TC, although putting m_{new} at different interior nodes of $\mathcal{T}$ will lead to different new TC tree structures, for any 2 different new TC trees $\mathcal{GT}_1$ and $\mathcal{GT}_2$, the identifiable links in $\mathcal{GT}_1$ and $\mathcal{GT}_2$ are the same.

Second, same link identifiability (with the existing and new monitors) leads to same $\mathcal{TEB}$, because i) the same set of *additional* identifiable links (due to the new monitor) create the same reduction on $\mathcal{TEB}$, ii) the total reduction on the error bounds for the remaining unidentifiable links is only affected by the additional identifiable links.

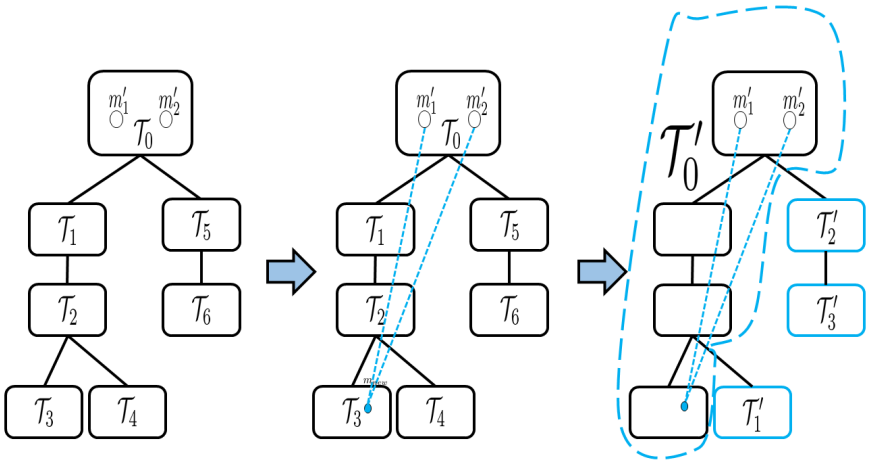
Therefore, we only need to determine the right TC to place m_{new}. In the following, we address this problem by analyzing how the new monitor would transform the original TC tree.

Adding one more monitor to $\mathcal{G}_{new}$ will transform the original TC tree into a new TC tree that is restructured by the virtual links between m_{new} and $\{m'_1, m'_2\}$. Use Fig. 4.5a as an example, there are 6 non-root TCs in the original TC tree, which all are the candidate TC for deploying m_{new}.

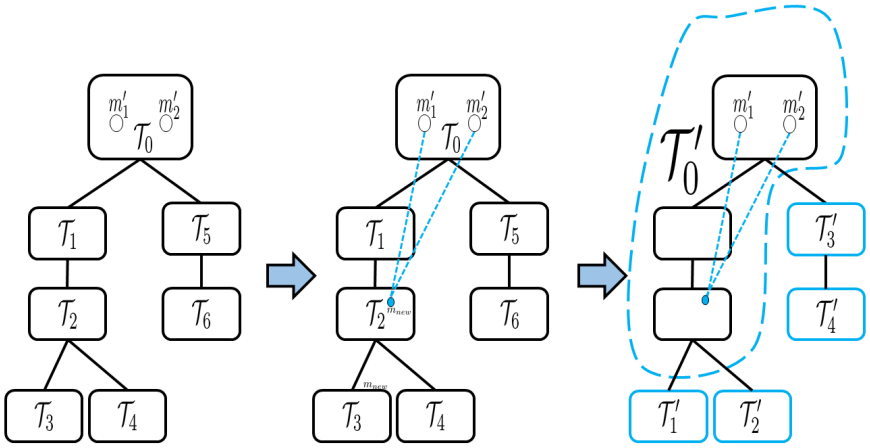
Remark 4. *$\mathcal{T}_0$ is ruled out from the candidate set because placing m_{new} into $\mathcal{T}_0$ does not change the original TC tree structure and thus will not make any difference on $\mathcal{TEB}$.*

We first consider the situation where the non-root TCs are all 3-vertex-connected TCs. For instance, if m_{new} is placed into $\mathcal{T}_3$, as shown in Fig 4.5a, the edges (*2-vertex-cut*s) between $\mathcal{T}_0$ and $\mathcal{T}_1$ ($e_{0,1}$), between $\mathcal{T}_1$ and $\mathcal{T}_2$ ($e_{1,2}$), and between $\mathcal{T}_2$ and $\mathcal{T}_3$ ($e_{2,3}$) will all disappear in the new TC tree because with the virtual links l_{m_{new},m'_1}, l_{m_{new},m'_2}, the *2-vertex-cut*s $e_{0,1}$, $e_{1,2}$, $e_{2,3}$ are no longer *2-vertex-cut*s in the new TC tree. Hence, $\mathcal{T}_0$, $\mathcal{T}_1$, $\mathcal{T}_2$ and $\mathcal{T}_3$ merge into $\mathcal{T}'_0$ in the new TC tree. But if m_{new} is placed into $\mathcal{T}_2$, as shown in Fig 4.5b, only $\mathcal{T}_0$, $\mathcal{T}_1$, and $\mathcal{T}_2$ are merged into $\mathcal{T}'_0$ in the new TC tree.

The other possible situation is that some non-root TCs are circle TCs. These circle TCs should appear along some tandem sub-tree. After m_{new} is placed into the leaf node of this tandem tree, such tandem sub-tree will not merge into $\mathcal{T}'_0$ in the new TC tree, because there may exist *2-b-c* links in a circle TC, which prevent this tandem tree to be 3-vertex-connected component in the new TC tree. For example, in Fig 4.6, there is a circle TC $\mathcal{T}_1$ in the leftmost original TC tree. After placing m_{new} in the leaf TC $\mathcal{T}_2$, say at node v_6, this tandem TC tree will not merge into $\mathcal{T}'_0$.



(a) Place m_{new} into $\mathcal{T}_3$.



(b) Place m_{new} into $\mathcal{T}_2$.

Figure 4.5: Examples for monitor placement.

4.4.2 Tandem TC Tree

In the above, we have disclosed the TC tree transform should a new monitor be added into a non-leaf TC. Next, we present a theorem for the deployment of new monitor when the TC tree is a tandem tree.

Theorem 12. *For a tandem TC tree with non-root nodes* $\{\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_n\}$ *(*$n \geq 1$*) in tandem (i.e.,* $\mathcal{T}_n$ *is the leaf TC on the TC tree), placing* m_{new} *at the interior node of* $\mathcal{T}_n$ *will maximally reduce the* $\mathcal{TEB}$*.*

Proof. After m_{new} is placed into some non-root TC, e.g., $\mathcal{T}_i$, edges (*2-vertex-cuts*) between $\mathcal{T}_j$ and $\mathcal{T}_{j-1}$ $(j = 1, 2, \cdots, i)$ are no longer edges in the new TC tree because of the virtual links $l_{m'_1, m_{new}}$ and $l_{m'_2, m_{new}}$ (Refer to Section 6.2.1 for detail when we discuss the classification of unidentifiable

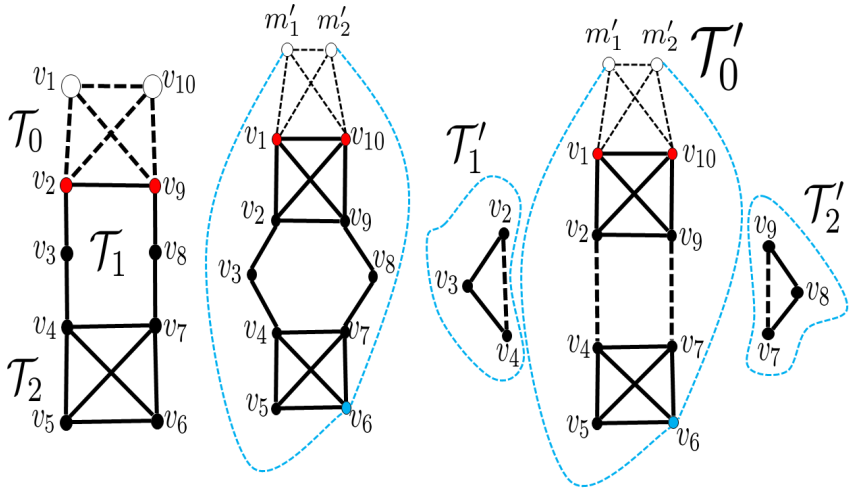


Figure 4.6: Example for monitor placement when non-root $\mathcal{T}_1$ is circle TC.

links). All the links in $\{\mathcal{T}_1, \cdots \mathcal{T}_j\}$ (except *circle TC links*) will belong to $\mathcal{T}_0'$ in the new TC tree, because i) any real link in $\mathcal{T}_0'$ is identifiable, their bound intervals will be reduced to 0, and ii) *circle TC links* cannot be in $\mathcal{T}_0'$ since they are *2-b-c* link in the new TC tree, as shown in Fig 4.6. Moreover, *circle TC links* in the new TC tree are also inside a circle TC and thus remain unidentifiable. The reduction of their bound intervals is only affected by the additional identifiable links in this tandem tree.

Therefore, for a tandem TC tree, placing m_{new} into $\mathcal{T}_n$ will create the *maximum set* of links whose identifiablility changes from unidentifiable to identifiable. Since no other placements can increase this maximum set, this placement leads to the maximal reduction on the remaining unidentifiable links. ■

Algorithm 5 MREB: Maximally Reducing $\mathcal{TEB}$

Input: network graph $\mathcal{G}$ and initial monitors deployment
Output: the location of m_{new} that maximally reduces $\mathcal{TEB}$

1: Obtain $\mathcal{G}_{new}$ and partition $\mathcal{G}_{new}$ into TCs recorded in a TC tree [13]
2: Put all the leaf node TCs in a set $\mathcal{L} = \{\mathcal{T}_l^1, \mathcal{T}_l^2, \cdots, \mathcal{T}_l^n\}$; also put their corresponding tandem trees in set $\mathcal{ST} = \{\mathcal{ST}_l^1, \mathcal{ST}_l^2, \cdots, \mathcal{ST}_l^n\}$
3: **for** each $\mathcal{ST}_l^i \in \mathcal{ST}$ **do**
4: **if** $\mathcal{ST}_l^i$ is *Independent* **then**
5: $\Delta\mathcal{TEB}^i, bAC = Get\Delta\mathcal{TEB}ForIndependentTree$
6: **else**
7: $\Delta\mathcal{TEB}^i = Get\Delta\mathcal{TEB}ForDependentTree$
8: $\Delta\mathcal{TEB}^j = \max(\Delta\mathcal{TEB}^1, \Delta\mathcal{TEB}^2, \cdots, \Delta\mathcal{TEB}^n)$
9: **return** a random interior node of $\mathcal{T}_l^j$

4.4.3 General TC Tree

We now analyze a general TC tree. A random TC tree could be decomposed into several tandem sub-trees, each from the root $\mathcal{T}_0$ all the way down to a leaf node. For instance, in Fig. 4.5a, this

general TC tree has 3 tandem sub-trees $\mathcal{ST}_1 = \{\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3\}$, $\mathcal{ST}_2 = \{\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_4\}$, $\mathcal{ST}_3 = \{\mathcal{T}_0, \mathcal{T}_5, \mathcal{T}_6\}$. Because of Theorem 12, leaf nodes $\mathcal{T}_3$, $\mathcal{T}_4$ and $\mathcal{T}_6$ are the candidate TCs for placing m_{new}. Topologically speaking, $\mathcal{ST}_3$ is *independent* since it shares no TC with other tandem sub-tree, $\mathcal{ST}_1$ (or $\mathcal{ST}_2$) is *dependent* (with other tandem sub-trees), because it shares TCs with $\mathcal{ST}_2$ ($\mathcal{ST}_1$).

The two types of tandem sub-trees require different ways of calculating the reduced total error bound: placing m_{new} into $\mathcal{T}_6$ could only affect the unidentifiable links in $\mathcal{ST}_3$, i.e., $\mathcal{T}_6$'s corresponding tandem sub-tree. Placing m_{new} into $\mathcal{T}_3$ (or $\mathcal{T}_4$), however, will not only affect its corresponding tandem sub-tree, but also other tandem sub-trees that share TC with it. For example, if m_{new} is put in $\mathcal{T}_3$, the $\mathcal{TEB}$ of unidentifiable links in $\mathcal{T}_4$ could also be reduced if some unidentifiable links become identifiable in the communal TCs of $\mathcal{ST}_1$ and $\mathcal{ST}_2$, i.e., $\mathcal{T}_1$ and $\mathcal{T}_2$.

Therefore, $\{\mathcal{ST}_1, \mathcal{ST}_2\}$ and $\{\mathcal{ST}_3\}$ should be classified into different types of tandem sub-trees, which need different ways of calculating the reduced total error bound:

1. *Dependent tandem sub-tree*: it shares one or multiple TCs with other tandem sub-tree(s).
2. *Independent tandem sub-tree*: it does not a share TC with any other tandem sub-tree.

Algorithm 6 Get$\Delta\mathcal{TEB}$ForDependentTree

Input: a tandem tree $\mathcal{ST}_l$ is a *Dependent Tree*
Output: the corresponding $\Delta\mathcal{TEB}$

```
1: Call Get ΔTEB For Independent Tree(ST_l) to obtain the second output, array A_accumulator
2: ΔTEB = 0
3: for each T_i along ST_l (from top to bottom, index-base starting with 1) do
4:   if A_accumulator[i] ≠ 0 then
5:     if A_accumulator[i − 1] = 0 (i > 1) or i = 1 then
6:       Call Get ΔTEB From All Branches(T_i, accu) to get ΔTEB_a
7:       ΔTEB += ΔTEB_a
8: return ΔTEB
```

Treating the above different cases, Algorithm 5 shows the best place for deploying the new monitor. For a general TC tree with several leaf TC nodes as $\{\mathcal{T}_{l_1}, \mathcal{T}_{l_2}, \cdots, \mathcal{T}_{l_2}\}$, each corresponding to a tandem sub-tree, placing m_{new} in $\mathcal{T}_{l_i}$ makes a series of unidentifiable links in its tandem sub-tree become identifiable, based on which the reduction on $\mathcal{TEB}$ can be calculated. The best place for deploying the new monitor is a leaf TC that maximally reduces the $\mathcal{TEB}$.

Remark 5. *The worst-case time complexity of Algorithm 5 is $O(\beta N_{\mathcal{T}})$, where $N_{\mathcal{T}}$ is the total number of TCs and β is the number of operations of calculating $\mathcal{TEB}$ as discussed in Remark 3. The second output of Algorithm 7 is prepared for Algorithm 6.*

Algorithm 7 Get$\Delta\mathcal{TEB}$ForIndependentTree

Input: a tandem tree $\mathcal{ST}_l$ is an *Independent Tree*

Output: the corresponding $\Delta\mathcal{TEB}$ and an array $A_{accumulator}$

Initialize an empty array $A_{accumulator}$
$accumulator = 0$
$\Delta\mathcal{TEB} = 0$
for each $\mathcal{T}$ along this tandem tree (from top to bottom) **do**
  if $accumulator \neq 0$ **then**
    $\Delta\mathcal{TEB}$ += $accumulator*$ number of unidentifiable links in $\mathcal{T}$
  if $\mathcal{T}$ is a circle TC **then**
    $accumulator$ *= 2
  else
    $\Delta\mathcal{TEB}$ += $\frac{1}{2} * \sum$ (length of all the unidentifiable links' bound intervals in $\mathcal{T}$)
    if $\mathcal{T}$ has 2-b-c links **then**
      Find the exterior link combination $\{e_1, e_2\}$ (e_1 is incident to one vantage of $\mathcal{T}$, e_2 is incident to the other vantage of $\mathcal{T}$) with the smallest bound interval
      $accumulator$ += $\frac{|\mathcal{B}(e_1)+\mathcal{B}(e_2)|}{2}$
    else
      $accumulator = 0$
  Append $accumulator$ to the end of array $A_{accumulator}$
return $\Delta\mathcal{TEB}$ and $A_{accumulator}$

Algorithm 8 Get$\Delta\mathcal{TEB}$FromAllBranches

Input: a TC node $\mathcal{T}$, an integer $accu$

Output: $\Delta\mathcal{TEB}$ from all affected Branches staring from $\mathcal{T}$

$\Delta\mathcal{TEB}_a = 0$
for each child of $\mathcal{T}$ ($\mathcal{T}_c$) **do**
  $\Delta\mathcal{TEB}_a$ += $accu*$ (number of unidentifiable links in $\mathcal{T}_c$)
  if $\mathcal{T}_c$ is a circle TC **then**
    $accu$ *= 2
    $R = Get\Delta\mathcal{TEB}FromAllBranches(\mathcal{T}_c, accu)$
    return $\Delta\mathcal{TEB}_a + R$
  else
    $\Delta\mathcal{TEB}_a$ += $\frac{1}{2}\sum$ (length of all the unidentifiable links' bound intervals in $\mathcal{T}_c$)
    if $\mathcal{T}_c$ has 2-b-c links **then**
      Find the exterior link combination $\{e_1, e_2\}$ (e_1 is incident to one vantage of $\mathcal{T}_c$, e_2 is incident to the other vantage of $\mathcal{T}_c$) with the smallest bound interval, i.e., $|\mathcal{B}(e_1) + \mathcal{B}(e_2)|$ is the smallest among all the possible exterior link combination in $\mathcal{T}_c$
      $accu$ += $\frac{|\mathcal{B}(e_1)+\mathcal{B}(e_2)|}{2}$
      $R = Get\Delta\mathcal{TEB}FromAllBranches(\mathcal{T}_c, accu)$
      return $\Delta\mathcal{TEB}_a + R$

4.5 Performance Evaluation

We test the performance of MREB with real-world ISP networks, using several representative autonomous system (AS) Internet topologies collected by the Rocketfuel [40] project. Among the four AS networks studied, two are from Europe, one from Australia, and one from US. Their size is also different to represent different scales of AS networks. The parameters of the selected 4 networks are shown in Table 4.1, where $|L|, |V|$, and $N_{\mathcal{T}}$ denote the number of links, the number of nodes, and the number of TCs, respectively.

Table 4.1: Parameters of AS Network Topology

ISP Name	$\|L\|$	$\|V\|$	$N_{\mathcal{T}}$
Ebone (Europe)	381	172	37
Tiscali (Europe)	404	240	52
Telstra (Australia)	758	318	50
AT&T (US)	2078	631	154

We compare MREB with the following two monitor deployment strategies:

- **Random**: Deploy the new monitor at a randomly-selected node, excluding the nodes that already have a monitor.
- **MAIL**: Deploy the new monitor at a node that maximizes the number of *additional* identifiable links.

For each AS network, we perform 50 rounds of test. In each round, we set the link delay as a random number in the range from 1 second to 30 seconds. We partition the network into bi-connected components (BCs), and in each BC, we randomly deploy a monitor. This way of deploying initial monitors is to spread the monitors across the whole network and avoid "bad" deployment that puts all initial monitors in one BC. In addition, spreading initial monitors in each BC instead of each TC will lead to a good number of unindentifiable links, because there is no need for bound-based performance tomography if all links are identifiable already. We then calculate $\mathcal{TEB}$ with the deployed monitors. After that, we run different algorithms, i.e., MREB, Random, and MAIL, to deployed a new monitor, and calculate the $\mathcal{TEB}$ again using all deployed monitors. By comparing the value change of $\mathcal{TEB}$ before and after the new monitor deployment, we know the value of $\mathcal{TEB}$ reduction due to the deployed new monitor.

We implemented the algorithms in C++ and performed the tests on a commodity desktop computer (CPU: Xeon E5-2620 V3, 6 core HT, 2.4 GHz, MEM: 32 GB, OS: Windows 10 Professional). The longest running time in all the tested scenarios is about 10 seconds.

The test results are shown in Fig. 4.7, Fig. 4.8, Fig. 4.9 and Fig. 4.10. We can see that MREB consistently leads to the highest $\mathcal{TEB}$ reduction in different ISP networks. Interestingly, the performance of MAIL is sensitive to network topology. Comparing Fig. 5.6 and Fig. 5.6, we can see that MAIL may work better or worse than Random, depending on the underlying network topology.

Random deployment has a relatively stable performance, but may suffer terribly if the network is big (e.g., in the AT&T network).

The results for the average $\mathcal{TEB}$ reduction over 50 tests in every AS network are shown in Table 4.2. From the results, MREB can improve Random w.r.t. average $\mathcal{TEB}$ reduction by up to 1587%, and can improve MAIL by up to 245%. Even if checked with the smallest improvement, MREB can still improve Random by 72%, and MAIL by 63%.

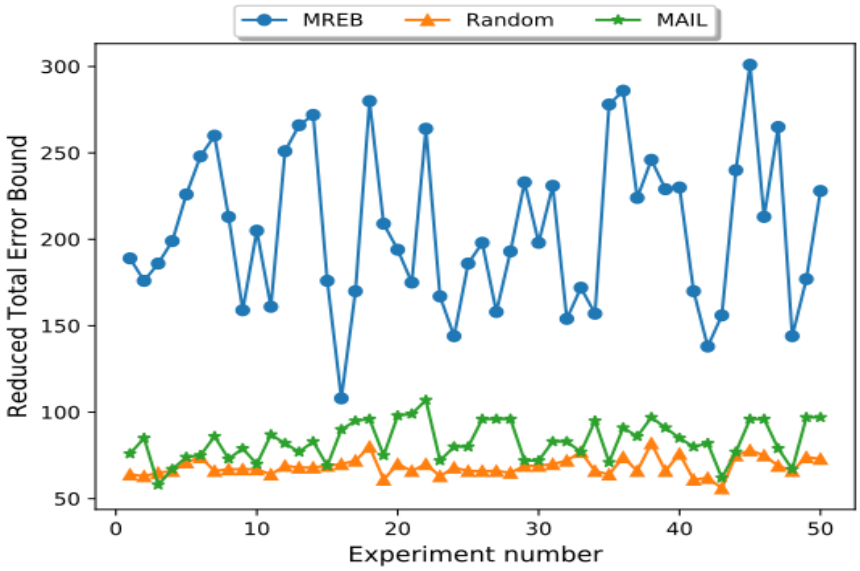


Figure 4.7: Performance of MREB, random, and MAIL on Ebone.

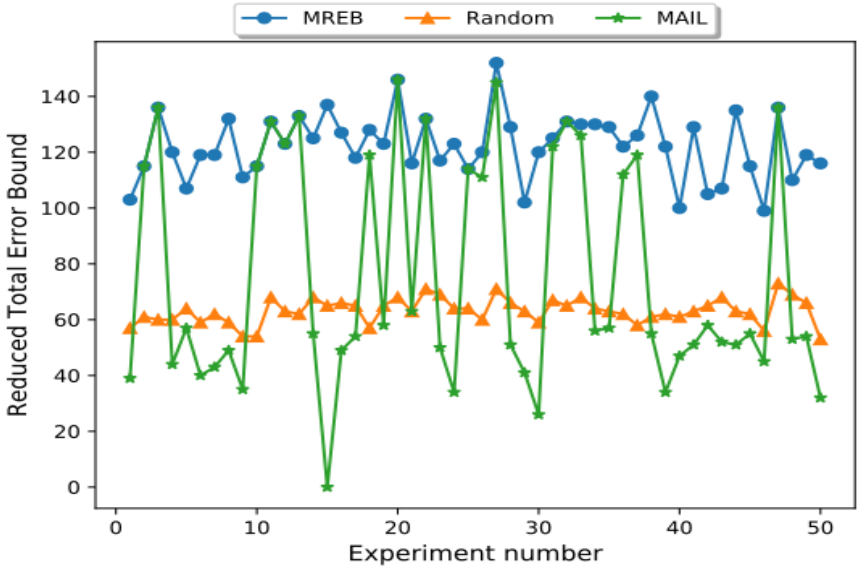


Figure 4.8: Performance of MREB, random, and MAIL on Tiscali.

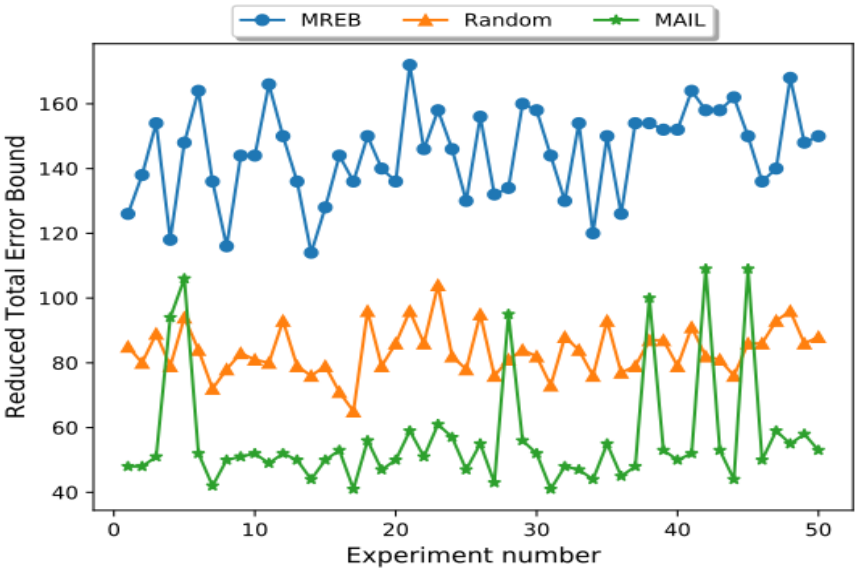


Figure 4.9: Performance of MREB, random, and MAIL on Telstra.

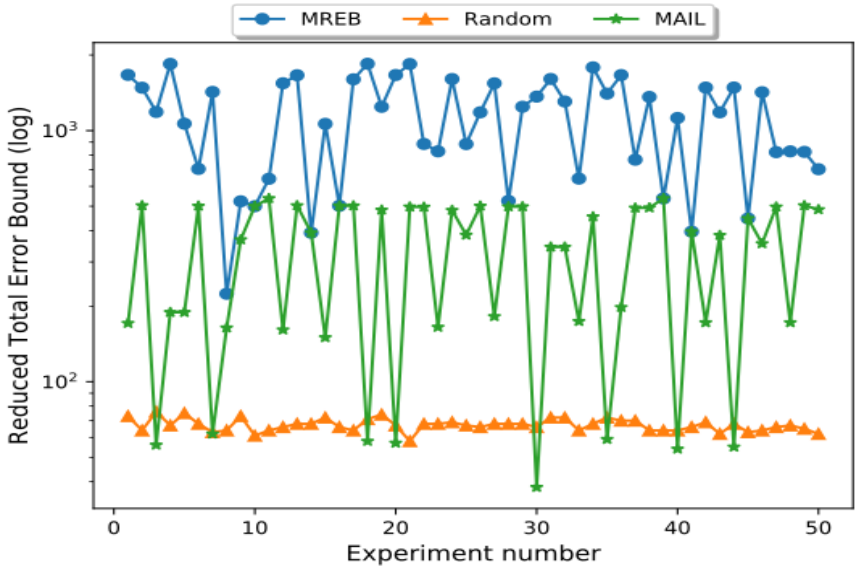


Figure 4.10: Performance of MREB, random, and MAIL on ATT.

Table 4.2: Average $\mathcal{TEB}$ Reduction over 50 Tests

Method / Topology	MREB Imp. over (Random, MAIL)	Random	MAIL
Ebone (Europe)	206 (198%, 148%)	69	83
Tiscali (Europe)	122 (93%, 63%)	63	75
Telstra (Australia)	145 (72%, 154%)	84	57
AT&T (US)	1130 (1587%, 245%)	67	328

Chapter 5

Controlling the Maximum Estimation Error in Bound-based Network Tomography

5.1 Overview

In Chapter 4, we aimed at controlling the total error bound in bound-based network tomography. In this chapter, we aim at developing an effective solution to minimize the maximum error bound ($\mathcal{MEB}$) over all the links in the network. To achieve this, we develop a method that theoretically guarantees (1) the minimum number of monitors required to bring down the $\mathcal{MEB}$ over all unidentifiable links, and (2) the best places where these new monitors should be deployed. Using this method repeatedly, we can push down the $\mathcal{MEB}$ gradually until the desired level is reached. With extensive simulation over real-world network topology, we demonstrate the effectiveness and robustness of our solution in reducing the maximum link error bound with network performance tomography.

5.2 Bound Reduction Analysis with Graph Decomposition

5.2.1 Graph Decomposition

First of all, it is worth mentioning that multiple links may have the same $\mathcal{MEB}$. For this reason, we should first answer the following two questions: (1) *what is the minimum number of* ***new*** *monitors needed to bring down the* $\mathcal{MEB}$ and (2) *where should we place these new monitors such that the* $\mathcal{MEB}$*s can be reduced by the maximum amount?* To answer the above questions, we use graph decomposition [19, 24, 31] to infer link performance.

As the first step, we follow the same procedure as in [31] that extends the original graph with two virtual monitors (m'_1, m'_2): the two virtual monitors only have virtual links to all physical monitors and a virtual link between themselves. The original problem of identifying link performance with the physical monitors is then transformed to an equivalent problem of identifying link performance with

the two virtual monitors [31]. Note that we do not need to care about the performance of virtual links and for bound error analysis we can simply assume the performance metric on the virtual links is all zero.

For the purpose of inferring link error bounds, the basic idea is to decompose the extended graph into TCs and the decomposed graph can be abstracted as a TC tree [19, 24], where a node in the tree represents a TC, the root of the tree is the TC consisting of the two virtual monitors, and the 2-vertex cut between two TCs as the "edge" between the two TC nodes. An example is shown in Fig. 5.4, where a graph (left figure) is decomposed into TCs (the middle figure), which is organized as a TC tree (the right figure). Note that we omit the details since decomposing a graph into TCs is a very basic problem, which has been well studied before [24] and could be found in many data-structure textbooks.

In the following, we denote l_m **as a link with the** $\mathcal{MEB}$, and $\mathcal{T}_m$ **as the TC that contains** l_m.

Remark 6. *Note that since the new monitors have not been deployed and new measurement results have not been collected, we cannot know the exact amount of error bound reduction. Nevertheless, we could know the best new monitor deployment strategy due to the special structure of TC tree. The best means (1) either the l_ms would become identifiable and in this case we reduce the $\mathcal{MEB}$ to zero, or (2) the l_ms would be still unidentifiable but the new monitor would reduce the $\mathcal{MEB}$ by the maximum amount (even if the exact amount is unknown).*

5.2.2 Simple Tandem TC Tree

It has been shown that placing a new monitor m_{new} at different *interior nodes* of TC $\mathcal{T}$ (i.e., nodes in $\mathcal{T}$ except vantage nodes) will result in the same link identifiability [19]. Following the same analysis on how a new monitor would transform the original TC tree [19], we first consider the most simple TC-tree structure, a tandem tree, as shown in Fig. 5.5. In such structure, only 1 new monitor is sufficient and necessary to reduce the $\mathcal{MEB}$:

Theorem 13. *For a tandem TC tree with non-root nodes $\{\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_n\}$ ($n \geq 1$) in tandem (i.e., $\mathcal{T}_n$ is the leaf TC of the TC tree). If l_ms appear in $\{\mathcal{T}_{m^1}, \mathcal{T}_{m^2}, \cdots, \mathcal{T}_{m^k}\}$ with $1 \leq m^1 < m^2 < \cdots < m^k \leq n$, placing one monitor m_{new} at an interior node of any $\mathcal{T}_{m^k+i}$ ($i = 1, 2, \cdots, n - m^k$) will reduce l_m's error bound by the same amount.*

Proof. In a tandem tree, after m_{new} is placed at an interior node of some non-root TC, e.g., $\mathcal{T}_i$, all the edges (i.e., *2-vertex-cuts*) between $\mathcal{T}_j$ and $\mathcal{T}_{j-1}$ ($j = 1, 2, \cdots, i$) are no longer edges in the new TC tree because of the virtual links $l_{m'_1, m_{new}}$ and $l_{m'_2, m_{new}}$. We use Fig. 5.1 and Fig. 5.2 as illustration examples. When m_{new} (in orange color) is placed into $\mathcal{T}_1$, the edge between $\mathcal{T}_0$ and $\mathcal{T}_1$ emerges into the new $\mathcal{T}'_0$ due to the virtual links (in orange dash lines); when m_{new} is placed into $\mathcal{T}_2$, all the edges in the old TC-tree emerge into the new $\mathcal{T}'_0$ for the same reason. Consequently, all the links in $\{\mathcal{T}_1, \cdots \mathcal{T}_j\}$ (except one case where there are *circle TC links* as shown in Fig. 5.3) will belong to $\mathcal{T}'_0$ in the new TC tree. The circle TC links do not belong to $\mathcal{T}'_0$ since they are *2-b-c* links in the new TC tree, as shown in Fig 5.3.

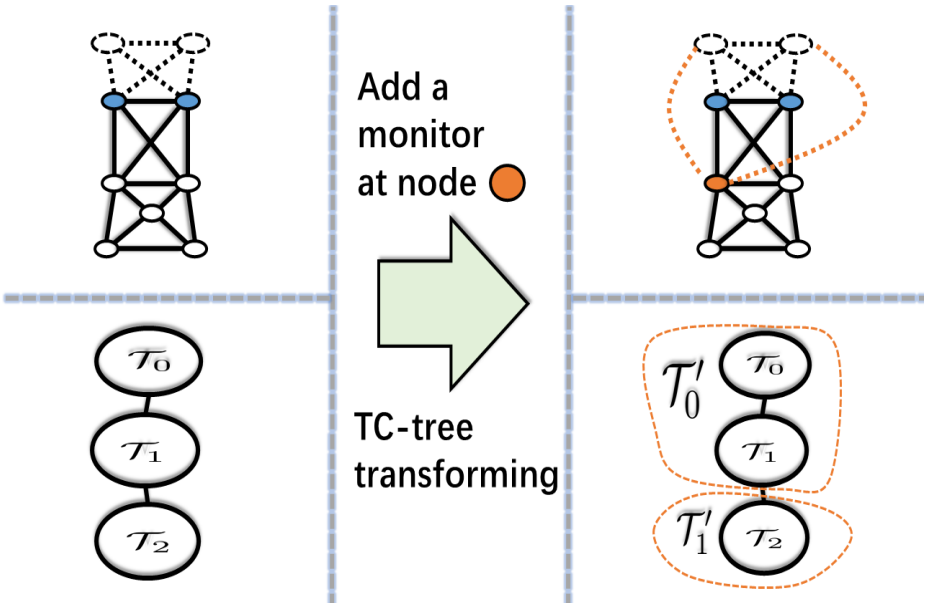


Figure 5.1: TC-tree transformation when m_{new} is put into $\mathcal{T}_1$.

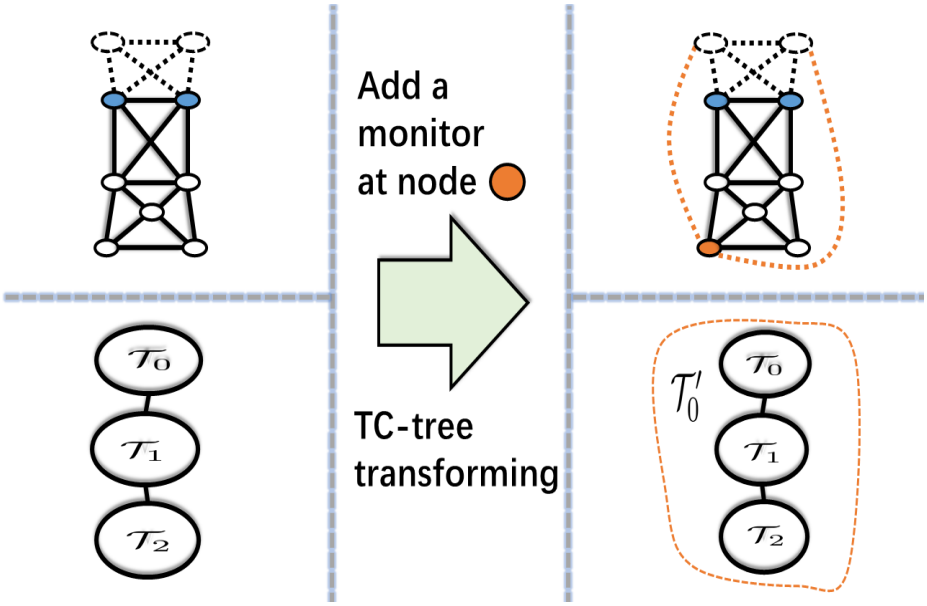


Figure 5.2: TC-tree transformation when m_{new} is put into $\mathcal{T}_2$.

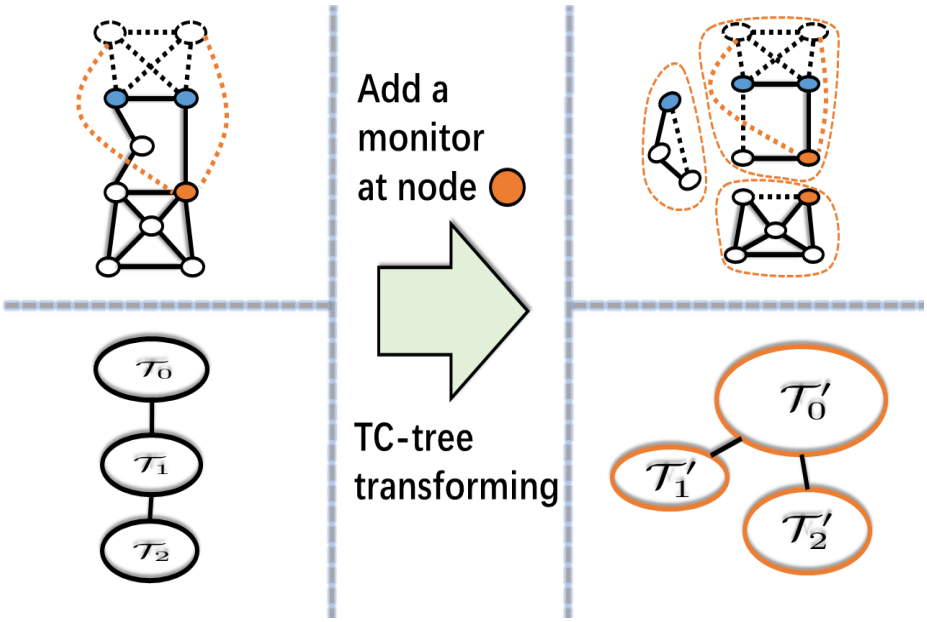


Figure 5.3: TC-tree transformation when m_{new} is put into $\mathcal{T}_2$, where $\mathcal{T}_1$ is a circle TC.

Therefore, deploying a new monitor in $\mathcal{T}_i$ will transform the existing TC-tree, and such a transformation will merge $\{\mathcal{T}_1, \cdots \mathcal{T}_i\}$ into the root TC $\mathcal{T}'_0$ in the new TC tree. As a result, (a) for any non-circle TC, its unidentifiable links are transformed into the root TC $\mathcal{T}'_0$ and thus become identifiable (i.e., their bound intervals will be reduced to 0), and (b) circle TC links in the new TC tree are also inside a circle TC thus remain unidentifiable. For a circle TC $\mathcal{T}_j$, although all of its link remain unidentifiable in the new TC tree, their bound interval will shrink as long as there are *newly* identifiable links (i.e., unidentifiable in old TC tree but identifiable in new TC tree) in $\{\mathcal{T}_1, \cdots \mathcal{T}_{j-1}\}$.

From the above analysis, we can see that by placing a new monitor at an interior node of any $\mathcal{T}_{m^k+i}$, l_m's bound would be reduced to 0 if l_m is in a non-circle TC or a smaller value (determined by the its ancestor TCs) if l_m is in a circle TC. Such a reduction only depends on the transformation of $\mathcal{T}_m$ and its ancestor TCs. Theorem 13 hence holds, because placing a monitor into any descendant TC of $\mathcal{T}_m$ will have the same effect on the transformation of $\mathcal{T}_m$ and its ancestor TCs. That is, placing a monitor into any $\mathcal{T}_m$'s descendant TC is equivalent w.r.t. reducing the amount of error bound on l_m. ■

Theorem 14. *For a tandem TC tree with non-root nodes $\{\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_n\}$ ($n \geq 1$) in tandem (i.e., $\mathcal{T}_n$ is the leaf TC of the TC tree). If l_ms appear in $\{\mathcal{T}_{m^1}, \mathcal{T}_{m^2}, \cdots, \mathcal{T}_{m^k}\}$ with $1 \leq m^1 < m^2 < \cdots < m^k \leq n$, placing a new monitor into $\mathcal{T}_i$ is no better than placing it into $\mathcal{T}_j$ ($i < j$) in the sense that the former reduces the error bound by an amount no larger than that of the latter.*

Proof. From the proof of Theorem 13, it is straightforward that placing a monitor into $\mathcal{T}_j$ will result in more (or at least equal) bound reduction on links than placing it into $\mathcal{T}_i$, where $i < j$. ■

Based on Theorems 13 and 14, in a single tandem tree with leaf TC $\mathcal{T}_n$, no matter where the l_m is, simply placing a new monitor at any interior node of $\mathcal{T}_n$ will reduce the $\mathcal{MEB}$ by the maximum amount.

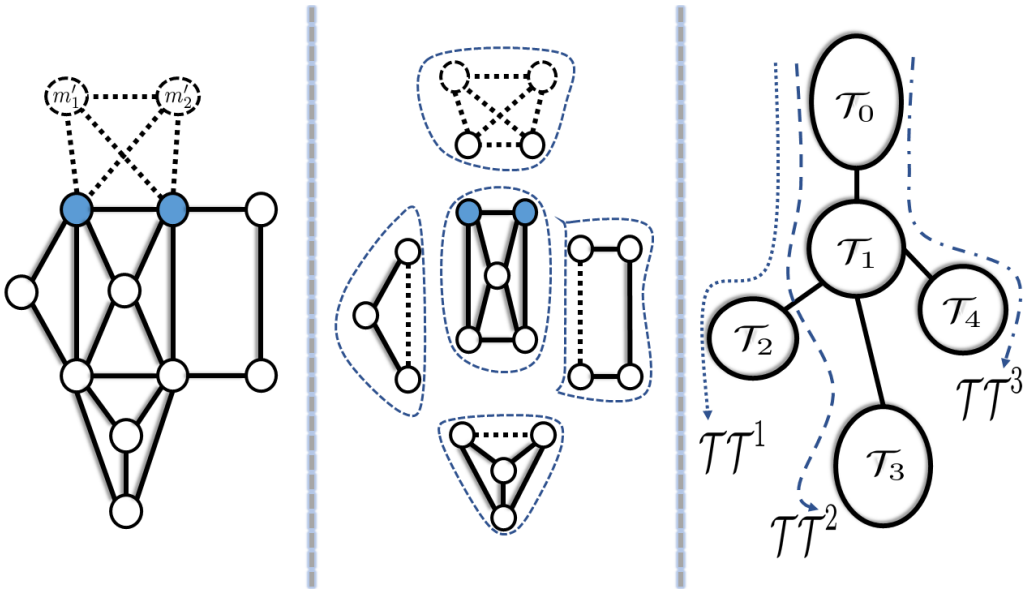


Figure 5.4: Graph decomposition into TCs.

5.2.3 General TC Tree

We then consider a more general scenario, where the TC-tree is not a simple linear tandem structure. Listing all paths from the root to the leaves, we can decompose the TC tree to several sub tandem trees, e.g., there are 3 sub tandem trees, $\mathcal{TT}^1 = \{\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2\}$, $\mathcal{TT}^2 = \{\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_3\}$, and $\mathcal{TT}^3 = \{\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_4\}$, in Fig. 5.4.

Case 1

These tandem trees are independent, meaning that $\mathcal{T}_0$ is the only ancestor node among these tandem trees. In this case, if a tandem tree contains an l_m, then we should deploy a monitor at an interior node of the leaf TC of the tandem tree.

Case 2

These tandem trees are not independent, meaning that some tandem trees share other ancestor nodes besides $\mathcal{T}_0$, as shown in Fig. 5.4. In this case, we need to consider two sub-cases:

Case 2.1: there is only one $\mathcal{T}_m$. Finding any one sub tandem tree that includes $\mathcal{T}_m$ and deploying a monitor at an interior node of the leaf TC of this tandem tree will reduce the $\mathcal{MEB}$ by the maximum amount based on Theorems 13 and 14.

Case 2.2: there are two or more $\mathcal{T}_m$'s. For *any* two $\mathcal{T}_m$'s:

- If they are located on the same sub tandem tree, deploying a monitor at an interior node of the leaf TC of this tandem tree will reduce the $\mathcal{MEB}$ by the maximum amount.
- If they are located on different but independent two sub tandem trees (i.e., the two sub tandem tree only have $\mathcal{T}_0$ as the ancestor), this case is the same as Case 1.

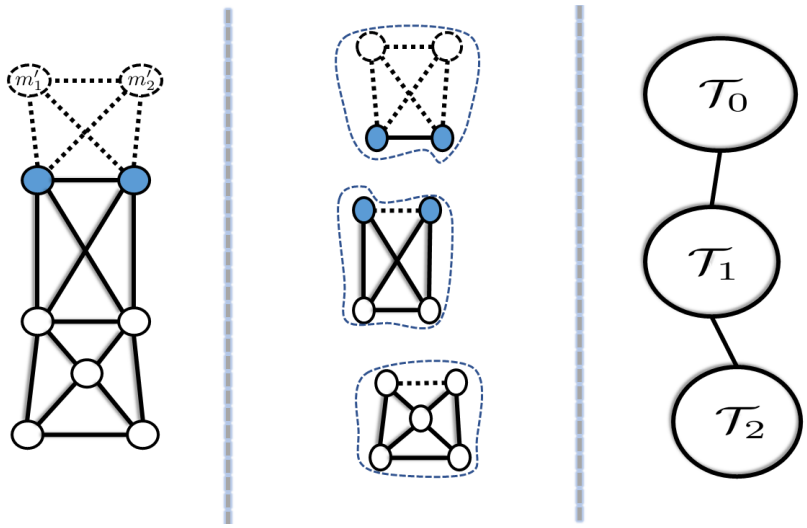


Figure 5.5: Tandem TC tree structure.

- (**Special Case**) If they are located on different and dependent two sub tandem trees (i.e., the two sub tandem trees have another ancestor, say $\mathcal{T}_a$, in addition to $\mathcal{T}_0$). From the above analysis, we know that at most deploying two new monitors, one at an interior node of the leaf TC of one sub tree and the other at an interior node of the leaf TC of the other sub tree, will bring down the $\mathcal{MEB}$ by the maximum amount.

Remark 7. *In the above special case, we may reduce the number of new monitors, with potentially losing theoretical guarantee on the maximum reduction on $\mathcal{MEB}$ (i.e., a trade-off). We use Fig. 5.4 to illustrate the scenario. Assume that $\mathcal{T}_2$ and $\mathcal{T}_3$ are both $\mathcal{T}_m$. If we place m_{new} in $\mathcal{T}_2$, we can reduce the error of l_m in $\mathcal{T}_2$. In addition, some unidentifiable links in $\mathcal{T}_1$ may become identifiable, since the new monitor may change the existing TC-tree structure. If these newly identifiable links are part of the measurement paths targeted to links in $\mathcal{T}_3$, error of l_m in $\mathcal{T}_3$ will shrink as well. If this happens, one monitor will suffice to reduce the l_ms in both $\mathcal{T}_2$ and $\mathcal{T}_3$. Nevertheless, the reduced amount on each l_m would be unknown unless they become identifiable. If the reduced amount on l_m is unknown, we have no way to know*[1] *whether deploying m_{new} at $\mathcal{T}_2$ is better than deploying it at $\mathcal{T}_3$, and thus we have no theoretical guarantee on the maximum reduction on the $\mathcal{MEB}$.*

With all the above analysis, we design Algorithm 9 (*MPMM*) that returns the minimum number of monitors and the candidate places where they could be deployed. The pseudo code is self-explanatory and reflects the analysis steps introduced above. The time complexity of Algorithm 9 is mainly on graph decomposition (Line 2), which is $O(|V| + |L|)$ where $|V|$ and $|L|$ are the number of nodes and the number of links of the graph, respectively [24].

[1]Note that we are not allowed to use trail-and-error to compare which one is better.

Algorithm 9 Minimum Monitor Placement for Maximum Reduction on $\mathcal{MEB}$ (MPMM)

Input: $\mathcal{G}$: the network; $\mathcal{L}_m$: the set of l_m links

Output: Locations for deploying new monitors

```
1: obtain G_new by adding two virtual monitors and corresponding virtual links [46];
2: partition G_new into tri-connected components, which are recorded in a TC tree [13, 24];
3: create an empty candidate TC set, CTS;
4: for each l_m in L_m do
5:    find the leaf TC along the longest tandem TC tree where this l_m belongs and denote this leaf
      TC as LTC; if this l_m corresponds to multiple longest tandem TC trees, record the multiple
      LTCs;
6:    for each LTC do
7:       if LTC ∈ CTS then
8:          break; /*do nothing since a monitor has already been deployed in LTC*/
9:       else
10:         find an interior node of LTC and label this node as a candidate location for monitor
            deployment;
11:         add LTC into CTS; /* i.e., this LTC covers l_m */
12: for each LTC in CTS do
13:   eliminate this LTC and the monitor inside if it only covers one l_m, which has been covered
      by another LTC in CTS; /*Remove redundant monitors, if any */
14: return all labelled candidate nodes;
```

5.3 Performance Evaluation

5.3.1 Performance of MPMM

We use 4 real-world ISP networks from the project Rocketfuel [40] to evaluate the performance of our algorithms. Table 5.1 lists the topological features of these networks, where where $|L|, |V|$, and $N_{\mathcal{T}}$ denote the number of links, the number of nodes, and the number of TCs, respectively.

Table 5.1: Parameters of AS Network Topology

ISP Name	$\|L\|$	$\|V\|$	$N_{\mathcal{T}}$
Ebone (Europe)	381	172	37
Tiscali (Europe)	404	240	52
Telstra (Australia)	758	318	50
AT&T (US)	2078	631	154

We perform 100 rounds of experiments in each network above. In each round, we randomly generate delay uniformly distributed in $[1, 100]$ on each link. Note that the delays are used *only* for determining the end-to-end delay for a measurement path. As the initial setup, we partition $\mathcal{G}$ into BCs, and in each BC, randomly deploy a monitor. With these initial monitors, we use SLM to

determine MPs, with which we use the method introduced in [19] to find the $\mathcal{MEB}$ and the link(s) (i.e., l_m's) that have the $\mathcal{MEB}$.

In Section 5.2, we have shown, *theoretically*, that MPMM can guarantee the maximum amount reduction on the error bounds of l_m's. To further demonstrate the advantages of MPMM, we adopt a "conservative" evaluation as follows: we use MPMM to deploy new monitors to reduce the $\mathcal{MEB}$, and with the newly deployed monitors, we re-build MPs with SLM and re-calculate the value of $\mathcal{MEB}$. We treat the difference between the previous $\mathcal{MEB}$ and the new $\mathcal{MEB}$ as the reduced amount on $\mathcal{MEB}$ (i.e., the reduced amount in the evaluation is measured as the $\mathcal{MEB}$ difference between two consecutive rounds rather than the absolute bound reduction on l_m's).

Remark 8. *We call the above evaluation strategy "conservative", because MPMM only aims at reducing the error bounds of l_m's with the maximum possible amount. The second largest error bound, after we apply MPMM and re-calculate the error bounds, now becomes the $\mathcal{MEB}$. It is possible that MPMM, while pushing down the error bounds of l_m's, has no impact on the second largest error bound, and thus our test method would lead to poor performance w.r.t. the reduced amount on $\mathcal{MEB}$ measured above. Nevertheless, as we will see, even in such "conservative" evaluation, MPMM has significantly better performance than other baseline methods (e.g., Random and MAIL defined below).*

As a comparison, we compare MPMM with the following baselines for new monitor deployment:

- Random: Deploy the new monitor at a randomly-selected node, excluding the nodes that already have a monitor.
- MAIL: Deploy the new monitor at a node that maximizes the number of *additional* identifiable links.

For a fair comparison, we set the number of new monitors added by Random and MAIL as the same as that by MPMM.

The test results are shown in Fig. 5.6, Fig. 5.7, Fig. 5.8, and Fig. 5.9. We can see that MPMM and MAIL consistently outperform Random. Actually, in most cases Random method cannot reduce the $\mathcal{MEB}$ at all. Comparing MPMM with MAIL, we can see that MPMM reduces a higher amount on $\mathcal{MEB}$ in most cases, although in some cases MAIL leads to a higher $\mathcal{MEB}$ reduction (refer to Remark 8 for the explanation). On average, **MPMM leads to about** 2**,** 2**,** 24**, and** 9 **times more $\mathcal{MEB}$ reduction than MAIL** in Ebone, Tiscali, Australia, and AT&T, respectively.

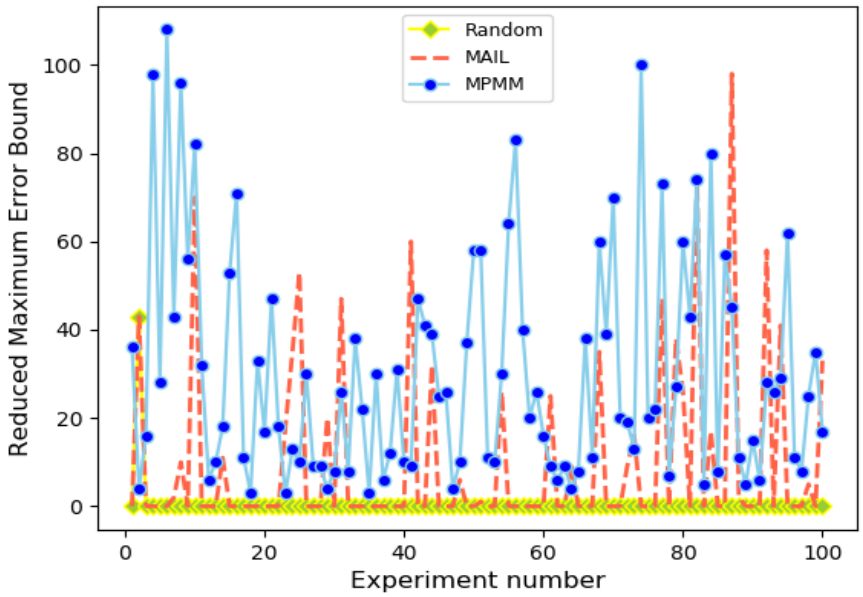


Figure 5.6: The reduced amount on $\mathcal{MEB}$ with MPMM, Random, and MAIL on Ebone

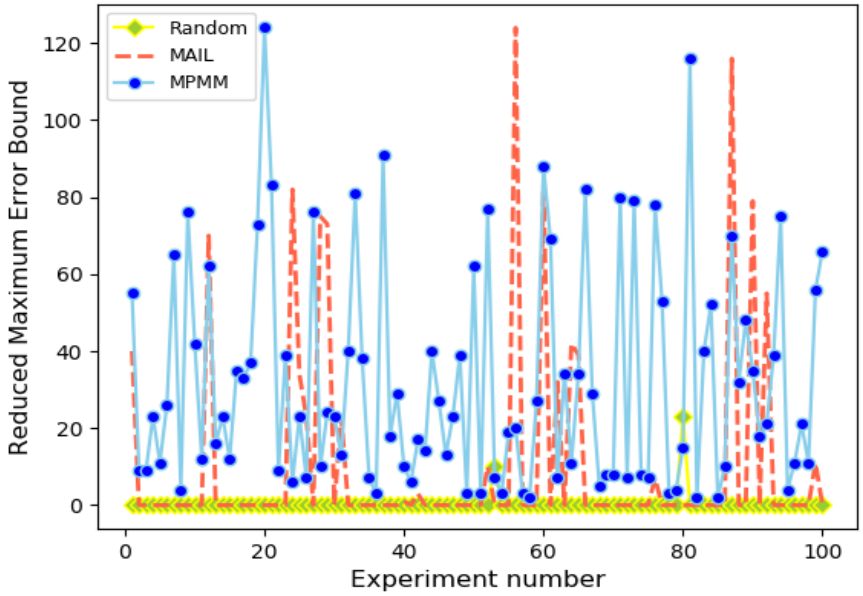


Figure 5.7: The reduced amount on $\mathcal{MEB}$ with MPMM, Random, and MAIL on Tiscali

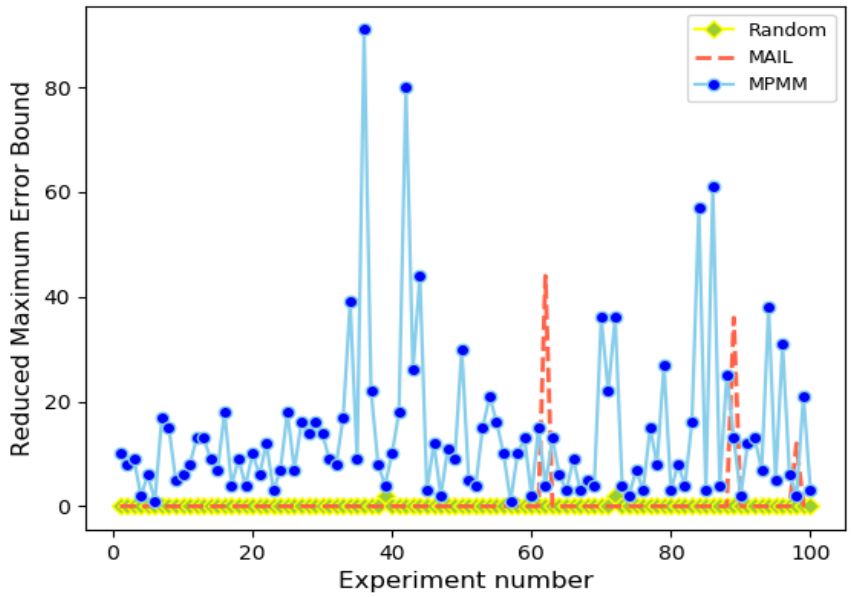


Figure 5.8: The reduced amount on $\mathcal{MEB}$ with MPMM, Random, and MAIL on Telstra

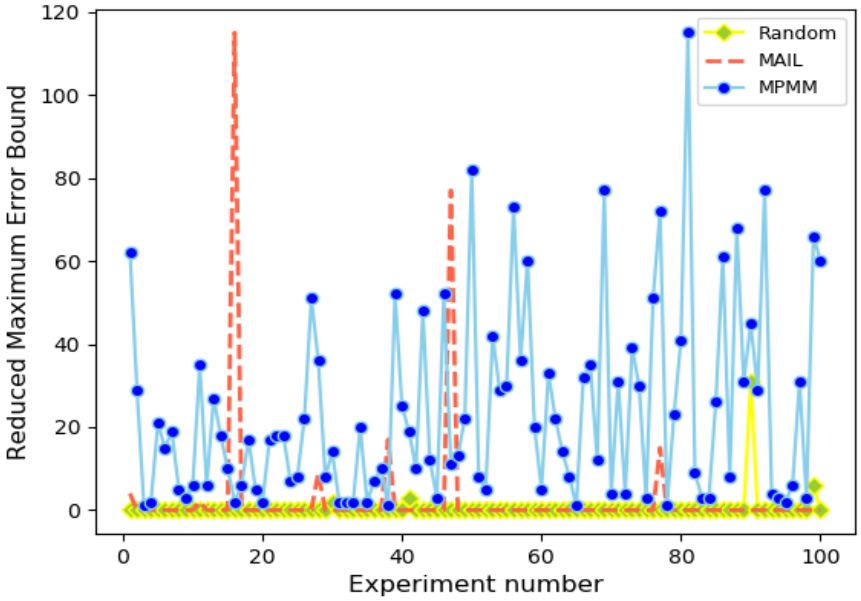


Figure 5.9: The reduced amount on $\mathcal{MEB}$ with MPMM, Random, and MAIL on AT&T

Chapter 6

Constructing the Least Measurement Paths

6.1 Overview

In the previous two Chapters, we investigated methods on controlling the total error bound and the maximum error bound in bound-based network tomography, respectively. A missing piece in these chapters is how to build measurement paths. We answer this question in this chapter.

We first propose an optimal path construction method (CMMP) for obtaining the tightest total error bound under bound-based tomography. It significantly reduces the total number of measurement paths required for deriving the tightest total error bound by only employing the *necessary and sufficient* MPs. What is worth noting is that CMMP constructs the minimum number of MPs with zero knowledge of any measurement beforehand and still guarantees the tightest total error bound over unidentifiable links. In addition, we design a sequentially learning-based measurement (SLM) to further reduce number of MPs with a relaxed "sequentially measurement" assumption.

6.2 Construct the Least Measurement Paths

Given a set of deployed monitors, we can form an *LSM* by listing all possible MPs and use the method in the Chapter 4 to obtain the tightest $\mathcal{TEB}$. Nevertheless, the total number of possible MPs is huge, and it is well known that listing MPs between two monitors is $\#P$-complete [43]. Naturally, we need to answer the following question: given a set of deployed monitors, what are the least MPs with which we can derive the tightest total error bound?

To answer this question, we first classify the unidentifiable links into three types (Section 6.2.1). Based on each type's special topological features, we build the MPs that are necessary and sufficient to infer the tightest error bounds of links in this type (Section 6.2.2).

6.2.1 Classifying Unidentifiable Links with Graph Decomposition

We first classify the identifiability of links, using the same technique of decomposing the network into tri-connected components (TCs) [31, 46]. We then obtain a much smaller set of MPs using the TC tree.

Given a network $\mathcal{G} =< V, L >$ with a pre-determined deployment of k ($k \geq 2$) monitors, the link identifiability problem could be converted to a 2-monitor problem by introducing two virtual monitors, each connecting to existing monitors with virtual links [31]. Following the same graph construction in [46], we can obtain the extended graph $\mathcal{G}_{new}$ with only 2 virtual monitors m'_1 and m'_2, in which $\mathcal{G}$ is the *interior graph* of $\mathcal{G}_{new}$. Next, we decompose $\mathcal{G}_{new}$ into TCs, which are recorded in a TC tree [13]. According to [46], the resulted TCs have the following topological properties:

1. The two virtual monitors, m'_1 and m'_2, are included in the same TC, denoted by $\mathcal{T}_0$, and all real monitors belong to $\mathcal{T}_0$;

2. Each TC $\mathcal{T}$ includes only two vantages. The vantages of $\mathcal{T}_0$ are m'_1 and m'_2, and each of the other TCs includes only one 2-vertex-cut that separates itself from m'_1 and m'_2.

3. All the TCs can be arranged in a *tree* structure, if we treat $\mathcal{T}_0$ as the *root*, the other TCs as *nodes*, and the 2-vertex-cut between the TCs as the *edge*. An example is shown in Fig. 6.1.

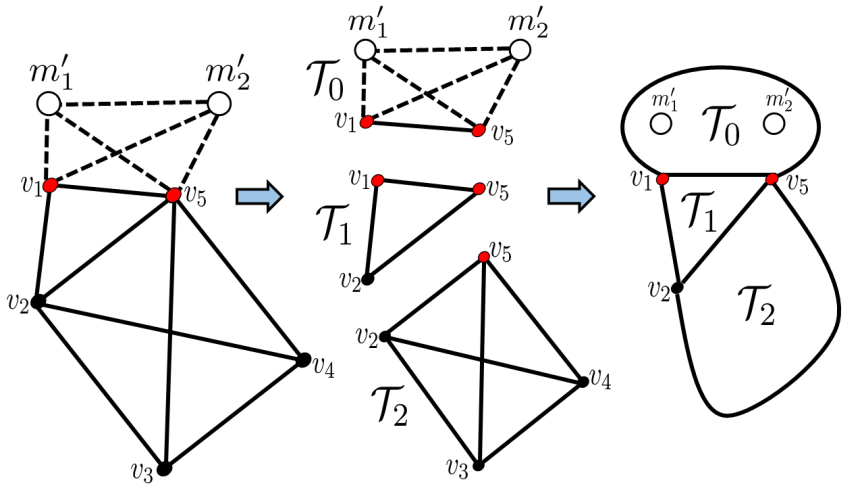


Figure 6.1: An example of graph decomposition. Note that v_1 and v_5 are the vantages w.r.t. $\mathcal{T}_1$, and v_2 and v_5 are the vantages w.r.t. $\mathcal{T}_2$.

Given a TC $\mathcal{T}$ with two vantages μ_1 and μ_2, denote $\widetilde{\mathcal{T}}$ as the subgraph of $\mathcal{T}$ obtained by removing the link between μ_1 and μ_2 (note that if there is no real link between μ_1 and μ_2, then a virtual link l_{μ_1,μ_2} was generated during the graph partition process). Based on the results for link identifiability [31, 46], every real link in $\mathcal{T}_0$ is identifiable, and only 3 types of links are unidentifiable in other TCs:

1. Exterior links of a 3-vertex-connected TC $\mathcal{T}_{3vc}$, i.e., links that are incident to only 1 vantage of the $\mathcal{T}_{3vc}$, are unidentifiable.

2. For a 3-vertex-connected TC $\mathcal{T}_{3vc}$, links of a 2-bridge-cut in $\widetilde{\mathcal{T}}_{3vc}$ are unidentifiable.

3. All the links in $\widetilde{\mathcal{T}}_{circle}$, where $\mathcal{T}_{circle}$ is a circle TC, are unidentifiable.

Remark 9. *The identifiability of the direct link between the two vantages can be determined in the parent TC which they belong to. As shown in Fig. 6.1, v_2 and v_5 are the vantages w.r.t. $\mathcal{T}_2$. The identifiability of l_{v_2,v_5} can be determined in $\mathcal{T}_1$, following the above procedure.*

6.2.2 Constructing the Least MPs

According to the above summary, there are only 3 types of links that are unidentifiable, therefore we need MPs that cover those links. To be specific, we need to construct a *necessary* (i.e., some unidentifiable links would not be covered if any MP in the set is removed) and *sufficient* (i.e., any other MP not in the set provides no extra information for determining the bounds of all unidentifiable links) *LSM* for the unidentifiable links.

After G_{new} is partitioned into TCs recorded in a TC tree, the three types of unidentifiable links are described with respect to a TC. Therefore, the MP construction consists of two major steps: In the first step, we consider how to obtain the *necessary* and *sufficient* MP segments **within** a specific TC $\mathcal{T}$. In the second step, we consider how to construct MP segments from any two monitors m_1, m_2 to $\mathcal{T}$.

Step 1: Constructing MP Segments inside a $\mathcal{T}$

When we only consider the parts of MP within a TC $\mathcal{T}$, the two vantages of $\mathcal{T}$ (μ_1 and μ_2) could be viewed as two monitors (since any MP needs to pass μ_1 and μ_2). We analyze based on the type of $\mathcal{T}$:

- **Case 1a:** when $\mathcal{T}$ is a $\mathcal{T}_{3vc}$ without *2-b-c* in $\widetilde{\mathcal{T}}_{3vc}$. The only unidentifiable links in $\mathcal{T}$ are the exterior links. Denote the two vantages of $\mathcal{T}$ as μ_1 (with n_{μ_1} exterior links incident to it) and μ_2 (with n_{μ_2} exterior links to it). Among all the possible segments of MPs that could be constructed inside $\mathcal{T}$, we only need $n_{\mu_1} \times n_{\mu_2}$ (necessary and sufficient) MP segments to obtain the tightest bound of unidentifiable links. This is because all the interior links of $\mathcal{T}$ are identifiable and thus no matter how a MP traverses in $\widetilde{\mathcal{T}}$, once the exterior links incident to the 2 vantages are selected (e.g., l_{μ_1} and l_{μ_2}), the MP segment in $\mathcal{T}$ will always give us an equation of the form:

  $$w_{l_{\mu_1}} + w_{\text{known metric}} + w_{l_{\mu_2}} = w$$

  which is equivalent to

  $$w_{l_{\mu_1}} + w_{l_{\mu_2}} = w - w_{\text{known metric}} = w'.$$

  Therefore, only one MP segment is enough because different MP trajectories in $\widetilde{\mathcal{T}}$ does not change the metric information about the two unidentifiable exterior links l_{μ_1} and l_{μ_2}. Since there are different combinations of exterior links, to obtain the complete metric information regarding those links, we need to include all the possible $n_{\mu_1} \times n_{\mu_2}$ combinations.

- **Case 1b:** when $\mathcal{T}$ is a $\mathcal{T}_{3vc}$ with *2-b-c* in $\widetilde{\mathcal{T}}_{3vc}$. In this case, besides the exterior links, the *2-b-c* links are also unidentifiable. Hence we need to take *2-b-c* links into account when constructing MP segments. By the nature of *2-b-c* links, any MP goes in $\mathcal{T}$ through μ_1 (μ_2) and goes out $\mathcal{T}$ through μ_2 (μ_1) must transverse **one and only one link** that belongs to a pair of *2-b-c* links. Therefore, if there are n_p pairs of *2-b-c* links, the total number of possible combination of unidentifiable links is

  $$2^{n_p} \times n_{\mu_1} \times n_{\mu_2},$$

  which is also the number of (necessary and sufficient) MP segments for obtaining the tightest bounds of unidentifiable links.

- **Case 1c:** when $\mathcal{T}$ is a $\mathcal{T}_{circle}$. Obviously there is only 1 possible MP (the circle itself) that goes in and out through $\mathcal{T}$'s vantages.

Remark 10. *In Case 1b, the situation where a 2-b-c link may also be an exterior link needs special care: it can only be counted as one link type, i.e., either exterior link or 2-b-c link, to avoid over-counting.*

Step 2: Constructing MP Segments from Monitors to $\mathcal{T}$

We next consider the MP segments between a pair of monitors (m_1 and m_2) and $\mathcal{T}$: any MP specially constructed for unidentifiable links (within $\mathcal{T}$) could be divided into 3 segments as

$$\text{MP} = P_{m_1,\mu_1}(P_{m_1,\mu_2}) + P_{\mu_1,\mu_2} + P_{\mu_2,m_2}(P_{\mu_1,m_2})$$

P_{m_1,μ_1} (P_{m_1,μ_2}) and P_{μ_2,m_2} (P_{μ_1,m_2}) are the parts of MP connecting monitors with $\mathcal{T}$ (i.e., $P_{m_1,\mu_1} + P_{\mu_2,m_2}$ or $P_{m_1,\mu_2} + P_{\mu_1,m_2}$). P_{μ_1,μ_2} denote the path in $\mathcal{T}$, as introduced in Step 1. In the following, we focus on ancestors of $\mathcal{T}$ because any MP connecting monitors with $\mathcal{T}$ must pass through them.

Denote the parent TC of $\mathcal{T}$ as $\mathcal{T}_p$, $\mathcal{T}_p$'s parent TC as $\mathcal{T}_{p-1}$, $\mathcal{T}_{p-1}$'s parent TC as $\mathcal{T}_{p-2}$, and so on. To ease discussion, we categorize TCs into normal and special TCs:

1. **Normal TC**: 3-vertex-connected TC $\mathcal{T}_{3vc}$ without *2-b-c* in $\widetilde{\mathcal{T}}_{3vc}$.
2. **Special TC**: i) 3-vertex-connected TC $\mathcal{T}_{3vc}$ with *2-b-c* in $\widetilde{\mathcal{T}}_{3vc}$ or ii) a circle TC.

Moreover, we classify the parent $\mathcal{T}_p$ as good or bad parent:

1. $\mathcal{T}_p$ is a **good parent** of $\mathcal{T}$ if $\mathcal{T}_p$ shares no vantage with $\mathcal{T}$.
2. otherwise, $\mathcal{T}_p$ is called a **bad parent** of $\mathcal{T}$ (i.e., $\mathcal{T}_p$ shares one vantage with $\mathcal{T}$).

Remark 11. *$\mathcal{T}_p$ can only share 1 vantage with its children, because if they share 2 vantages, they cannot form the "child-parent" relation in the tandem tree). In Fig. 6.1, $\mathcal{T}_1$ is a bad parent of $\mathcal{T}_2$, because they both have v_5 as their vantage.*

For a target TC $\mathcal{T}$, an MP connecting two monitors and its 2 vantages ($P_{m_1,\mu_1} + P_{\mu_2,m_2}$) must go through $\mathcal{T}_p$. Moreover, $\mathcal{T}_p$'s property could determine the identifiability of $P_{m_1,\mu_1} + P_{\mu_2,m_2}$: if and only if $\mathcal{T}_p$ is *normal* and is a *good parent*, the value of $P_{m_1,\mu_1} + P_{\mu_2,m_2}$ could be determined with

one MP. From now on, we call a parent TC **ideal TC** if it is *normal* and is a *good parent*; otherwise it is a **non-ideal TC**. Table 6.1 shows the detailed relationship between $\mathcal{T}_p$ and identifiability of $P_{m_1,\mu_1} + P_{\mu_2,m_2}$.

Table 6.1: Four Cases of $\mathcal{T}_p$

$\mathcal{T}_p$	$P_{m_1,\mu_1} + P_{\mu_2,m_2}$
2a: Normal + Good parent = Ideal	identifiable
2b: Normal + Bad parent = Non-ideal	unidentifiable
2c: Special + Good parent = Non-ideal	unidentifiable
2d: Special + Bad parent = Non-ideal	unidentifiable

We next discuss the 4 scenarios in Table 6.1. For a target TC $\mathcal{T}$ with 2 vantages μ_1 and μ_2, and its parent TC $\mathcal{T}_p$:

- **Case 2a:** we could obtain the value of segments $P_{m_1,\mu_1} + P_{\mu_2,m_2}$ in an MP based on the topological feature of this case: As shown in Fig 6.2, because $\mathcal{T}_p$ is normal TC and does not share vantage with $\mathcal{T}$, we can construct an MP going through $\mathcal{T}_p$ as $\text{MP}_f = P_{m_1,\mu_1} + l_{\mu_1,\mu_2}(p_{\mu_1,\mu_2}) + P_{\mu_2,m_2}$ (if there is no direct real link between μ_1 and μ_2, a path inside $\mathcal{T}_p$ between μ_1 and μ_2 could substitute l_{μ_1,μ_2}). Since l_{μ_1,μ_2} is an identifiable interior link of $\mathcal{T}_p$, subtracting its value from the measurement value of MP_f gives us the value of $P_{m_1,\mu_1} + P_{\mu_2,m_2}$. Note that obtaining the value of $P_{m_1,\mu_1} + P_{\mu_2,m_2}$ in such a way will need to build one MP (i.e., MP_f). After that, $P_{m_1,\mu_1} + P_{\mu_2,m_2}$ can be used to concatenate the segments inside $\mathcal{T}$ (built in Step 1) to obtain the minimal MP set for unidentifiable links in $\mathcal{T}$.

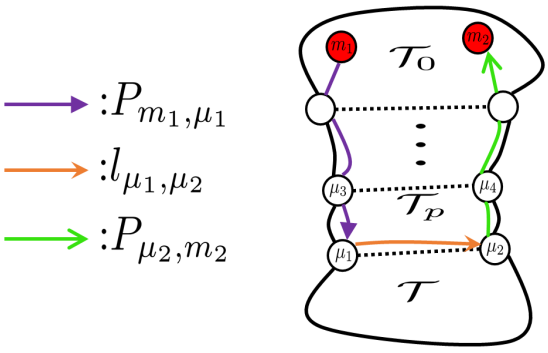


Figure 6.2: An example of Case 2a.

- **Case 2b:** when $\mathcal{T}_p$ is **normal** but a **bad parent**, as shown in Fig 6.3. Now the value of $P_{m_1,\mu_1} + P_{\mu_2,m_2}$ cannot be obtained, because μ_2 is also the vantage point of $\mathcal{T}_p$ and thus l_{μ_1}, μ_2 (p_{μ_1,μ_2}) is an unidentifiable exterior link (path) in $\mathcal{T}_p$. Thus, in order to include the path

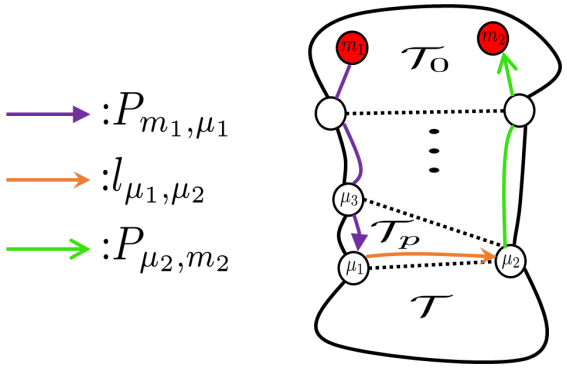


Figure 6.3: An example of Case 2b.

that has the minimum value, we need to construct n_{μ_1} (number of exterior links incident to μ_1) segments within $\mathcal{T}_p$.

- **Case 2c and 2d:** when $\mathcal{T}_p$ is **special**, no matter it is a **good parent** or **bad parent**, we cannot derive the value of $P_{m_1,\mu_1} + P_{\mu_2,m_2}$ for the same reason that l_{μ_1,μ_2} (p_{μ_1,μ_2}) is unidentifiable. For a circle TC, the unidentifiability of l_{μ_1,μ_2} (p_{μ_1,μ_2}) is straightforward because every link in a circle TC is unidentifiable [46]. For a $\mathcal{T}_{3vc}$ with *2-b-c* links, the unidentifiability of l_{μ_1,μ_2} (p_{μ_1,μ_2}) is caused by the fact that any p_{μ_1,μ_2} in this TC must pass through the *2-b-c* links, whose unidentifiability makes the whole segment unidentifiable. In this case, we have to construct different p_{μ_3,μ_1} containing all the exterior links, as shown in Fig. 6.4.

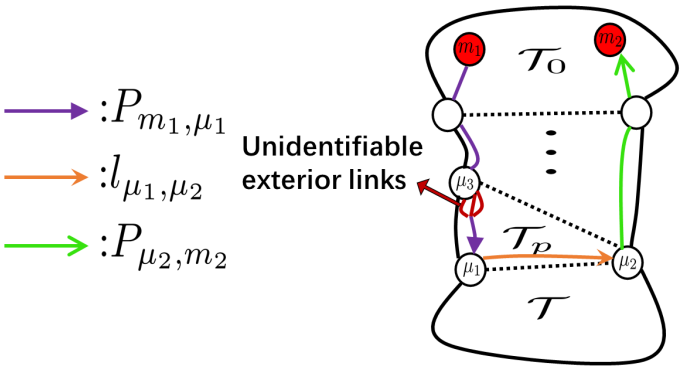


Figure 6.4: An example of Case 2c/2d.

Looking at the whole TC tree, $\mathcal{T}_p$ may have multiple child TCs. In Cases 2b, 2c, and 2d, we need a special care when $\mathcal{T}_p$ has multiple child TCs and $\mathcal{T}_p$ shares a vantage with $\mathcal{T}$'s sibling $\mathcal{T}_s$, as shown in Fig. 6.5. In this case, MP segments entering $\mathcal{T}$ through $\mathcal{T}_s$ may have a smaller value than the MP segments entering $\mathcal{T}$ via $\mathcal{T}_p$. Because of the zero knowledge of their real value beforehand, we should include all such possible MP trajectories. In this special case, we call $\mathcal{T}_p$ **reducible** because

a path entering $\mathcal{T}$ through $\mathcal{T}_s$ has a smaller value than a path entering $\mathcal{T}$ through $\mathcal{T}_p$.

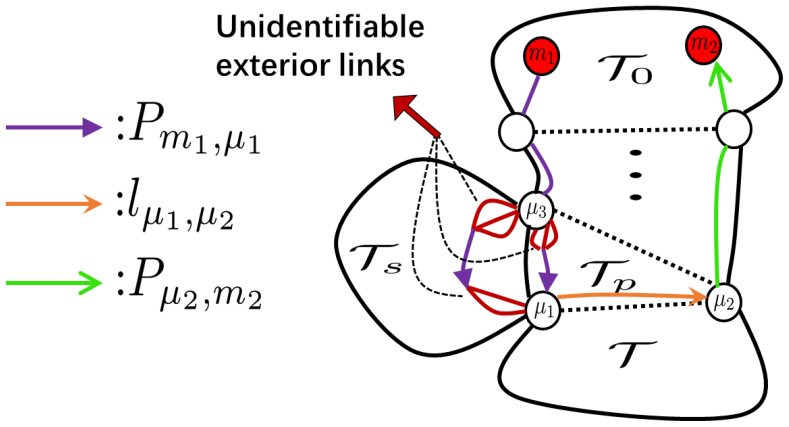


Figure 6.5: A non-ideal $\mathcal{T}_p$ is reducible when it has multiple child TCs ($\mathcal{T}$ and $\mathcal{T}_s$). It is possible that a path entering $\mathcal{T}$ through $\mathcal{T}_s$ has a smaller value than a path entering $\mathcal{T}$ through $\mathcal{T}_p$.

Put All Together

Algorithm 10 Constructing Minimal Measurement Paths (CMMP)

Input: a network graph $\mathcal{G}$ and deployed monitors

Output: minimal measurement paths

1: Obtain $\mathcal{G}_{new}$ by adding two virtual monitors and corresponding virtual links [31]
2: Partition $\mathcal{G}_{new}$ into tri-connected components, which are recorded in a TC tree [13]
3: Create an empty set $\mathcal{MP}$
4: **for** each non-root TC $\mathcal{T}_i$ **do**
5: Mark the 2 vantages of $\mathcal{T}_i$ as $\{\mu_1, \mu_2\}$
6: Call $FindPathsInsideTC(\mathcal{T}_i)$ and record the returned paths in $\mathcal{P}_i$
7: **for** any monitor pair (m_1, m_2) **do**
8: Identify an one-to-one, monitor-vantage mapping, denoted as (m_1, u_1) and (m_2, u_2) without loss of generality
9: Call $FindPathsFromMonitorToTC(\mathcal{T}_i)$ and record the returned paths from m_1 to u_1 in $\mathcal{P}_{m_1,\mu_1}$ and paths from m_2 to u_2 in $\mathcal{P}_{m_2,\mu_2}$
10: Construct an MP by concatenating three paths from the sets $\mathcal{P}_{m_1,\mu_1}$, $\mathcal{P}_i$, and $\mathcal{P}_{m_2,\mu_2}$, respectively, and construct all MPs by finding all possible path concatenations
11: Put all the constructed MPs into $\mathcal{MP}$
12: **return** $\mathcal{MP}$

Having known how to extract necessary and sufficient segments in all the scenarios listed in Table 6.1, we now study a branch of TCs along the TC tree: For a targeted TC $\mathcal{T}$, if its parent TC $\mathcal{T}_p$ is an ideal TC, then we construct an MP segment from two monitors to $\mathcal{T}$ according to Case 2a.

Nevertheless, if its parent TC $\mathcal{T}_p$ is non-ideal TC, we continue to look up to $\mathcal{T}_p$'s parent $\mathcal{T}_{p-1}$ (with vantages μ_3, μ_4) and **recursively** call the same procedure. That is, if $\mathcal{T}_{p-1}$ is an ideal TC, then the value of $P_{m_1,\mu_3} + P_{\mu_2,m_4}$ could be returned by the way described in Case 2a; if not, we look up $\mathcal{T}_{p-1}$'s parent until we reach the first ideal ancestor of $\mathcal{T}$, denoted as $\mathcal{T}_a$, which can return the value of the MP segment from two monitors to $\mathcal{T}_a$. Since we have analyzed all intermediate TCs between $\mathcal{T}_a$ and $\mathcal{T}$ (i.e., a tandem tree branch), we are able to construct a (necessary and sufficient) set of MPs that can be used to derive the tighest bounds for unidentifiable links.

According to the above analysis, the pseudo-code of constructing the least MPs is given in Algorithm 10. In Line 7 of Algorithm 10, all monitors are in the root TC, $\mathcal{T}_0$, as discussed in Section 6.2.1. In Line 10, two paths can be concatenated only if they share a common end node. Algorithm 10 needs to call two functions $FindPathsInsideTC$, which corresponds to Step 1, and $FindPathsFromMonitorToTC$, which corresponds to Step 2. Fig. 6.6 shows an example about how Algorithm 10 can reduce the number of MPs.

Algorithm 11 FindPathsInsideTC

Input: a TC $\mathcal{T}_i$ with two vantages denoted by $\{\mu_1, \mu_2\}$

Output: the set of paths inside $\mathcal{T}$ that will be used to construct an MP

1: Create an empty set $\mathcal{P}_i$
2: **if** $\mathcal{T}_i$ is a circle TC **then**
3: Concatenate all the links in $\mathcal{T}_i$ except the direct link between μ_1 and μ_2 (if exists), and put the path in $\mathcal{P}_i$
4: **else**
5: **for** each exterior link a_i incident to μ_1 **do**
6: **for** each *2-b-c* link b_j (if exists and is not an exterior link) **do**
7: **for** each exterior link c_k incident to μ_2 **do**
8: Find a path between μ_1 and μ_2 (within $\mathcal{T}_i$) containing a_i, b_j and c_k, put the path in $\mathcal{P}_i$
9: **return** $\mathcal{P}_i$

Algorithm 12 FindPathsForNonidealTC

Input: a non-ideal TC $\mathcal{T}$ and its two vantages $\{\mu_1, \mu_2\}$

Output: its associative path segments

1: **while** $\mathcal{T}$ is a bad parent **do**
2: **for** each $\mathcal{T}$'s child $\mathcal{T}_c$ that shares vantage with $\mathcal{T}$ **do**
3: Call $FindPathsForNonidealTC(\mathcal{T}_c)$
4: **return** $FindPathsInsideTC(\mathcal{T})$

Remark 12. *Partitioning the network into TCs can be done in linear time [24]. If we ignore Case 1b (i.e., $\widetilde{\mathcal{T}}_{3vc}$ has multiple 2-b-c), the worst-case time complexity of Algorithm 10 is $O(\alpha h N_{\mathcal{T}})$, where α is the maximum TC size among all the TCs, $N_{\mathcal{T}}$ is the total number of TCs, and h is the height of TC tree. In other words, it is polynomial of the number of links. Case 1b has only theoretical meaning and is listed only for the theoretical completeness of the paper. In practice, Case 1b can be easily*

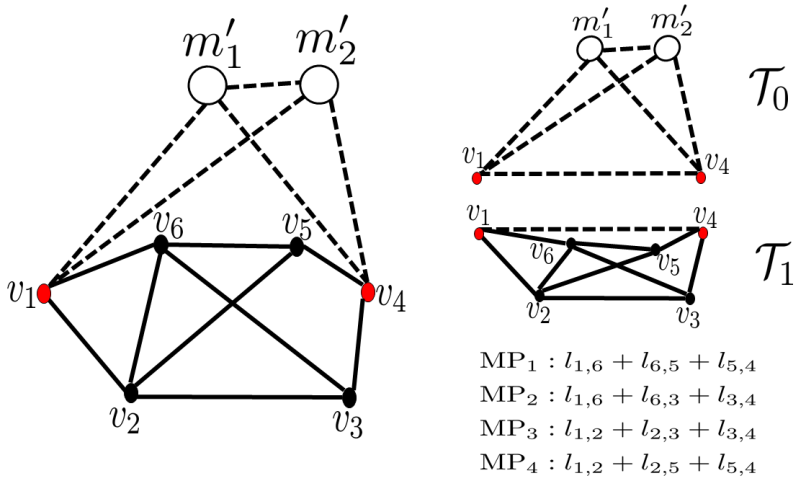


Figure 6.6: An example: there are 12 possible MPs between the two monitors (in red), whereas with Algorithm 10, only $2 \times 2 = 4$ MPs are needed to obtain the tightest $\mathcal{TEB}$.

avoided with proper initial deployment of monitors, e.g., deploying a monitor in each bi-connected component (BC) as suggested in Section 4.5.

Algorithm 13 FindPathsFromMonitorToTC

Input: a TC $\mathcal{T}_i$ (with 2 vantages μ_1 and μ_2) and two monitors m_1, m_2
Output: $\mathcal{P}_{m_1,\mu_1}$ that contains paths from m_1 to u_1, $\mathcal{P}_{m_2,\mu_2}$ that contains paths from m_2 to u_2

1: Find the first ancestor TC of $\mathcal{T}_i$ which is an **ideal TC**, denoted as $\mathcal{T}_a$ (with 2 vantages μ_a^1 and μ_a^2), put the sequence of consecutive TCs from $\mathcal{T}_i$ to the $\mathcal{T}_a$ in a set as $\mathcal{AS}_i = \{\mathcal{T}_i, \mathcal{T}_{i-1}, \mathcal{T}_{i-2}, \cdots, \mathcal{T}_{i-n}(= \mathcal{T}_a)\}$
2: Find a path from m_1 to μ_a^1 passing along the TCs from $\mathcal{T}_0$ to $\mathcal{T}_a$ and avoiding virtual links; put the path in set $\mathcal{P}_{m_1,\mu_1}$
3: Find a path (vertex-disjoint to the path in $\mathcal{P}_{m_1,\mu_1}$) from m_2 to μ_a^2 passing along the TCs from $\mathcal{T}_0$ to $\mathcal{T}_a$ and avoiding virtual links; put the path in set $\mathcal{P}_{m_2,\mu_2}$
4: **for** each TC $\mathcal{T}_{i-j}$ in $\mathcal{AS}_i$ **do**
5: Call $FindPathsInsideTC(\mathcal{T}_{i-j})$
6: **if** $\mathcal{T}_{i-j}$ is reducible **then**
7: **for** each TC $\mathcal{T}_{i-j}$'s child $\mathcal{T}_s$ that shares vantage with $\mathcal{T}_{i-j}$ **do**
8: Call $FindPathsForNonidealTC(\mathcal{T}_s)$
9: Construct $\mathcal{P}_{m_1,\mu_1}$ and $\mathcal{P}_{m_2,\mu_2}$ by concatenating all possible path combinations
10: **return** $\mathcal{P}_{m_1,\mu_1}$ and $\mathcal{P}_{m_2,\mu_2}$

6.3 Sequential Learning-based Measurement with Predetermined Monitors

6.3.1 One Batch Measurement v.s. Sequential Learning-Based Measurement

In the previous section, CMMP gives the minimum number of MPs under the assumption that all required MPs are constructed and used at the same time [19]. However, we find that if we relax the assumption in [19] and adopt a sequential measurement method, we can further reduce the number of MPs to achieve the same goal. The intuition is that, after we build several initial paths and collect measurement information with the paths, we gain more information on the link performance, which can be utilized to guide our construction of new MPs. We call this MP construction method as *sequential learning-based measurement (SLM)*. As a comparison, we refer the method in [19] as *one batch measurement (OBM)*.

6.3.2 Why Is SLM Possible?

As presented in Section 5.2, the TC-tree structure provides us with enough topological information to build MPs.

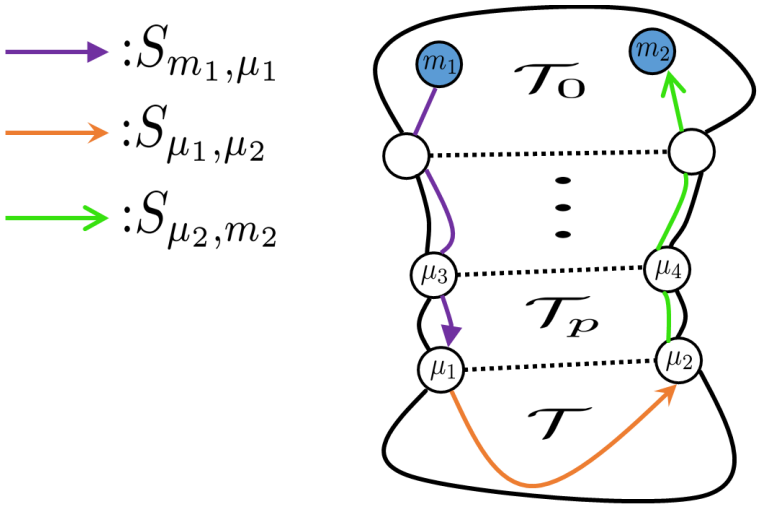


Figure 6.7: Topological illustration of an MP passing $\mathcal{T}$.

After we decompose the graph and build the TC tree, following the analysis in [19], there are 3 types of unidentifiable links in a TC, and any MP containing unidentifiable links in a TC $\mathcal{T}$ (with 2 vantages μ_1 and μ_2) must be comprised with three parts (as shown in Fig. 6.7):

$$\text{MP} = S_{m_1,\mu_1} + S_{\mu_1,\mu_2} + S_{\mu_2,m_2}$$

where $+$ means concatenation, S_{m_1,μ_1} is the segment from one virtual monitor (m_1) to an vantage of $\mathcal{T}$ (μ_1), S_{μ_1,μ_2} is the segment within $\mathcal{T}$, and S_{μ_2,m_2} is the segment from the other vantage of $\mathcal{T}$

(μ_2) to the other virtual monitor (m_2). Therefore, any MP that is not formed by these 3 segments can be excluded (from the perspectives of deriving the tightest bound of unidentifiable links in $\mathcal{T}$): because either it does not contain the targeted unidentifiable links in $\mathcal{T}$, or it travels longer distance in the network and thus contains more unnecessary unidentifiable segments in the linear equation, which results in looser error bounds.

Clearly, viewed from TC $\mathcal{T}$, the number of MPs depends on the possible combinations of the three path segments. The number of combinations may be reduced if we know more information about $\mathcal{T}$'s ancestor TCs via SLM. Such reduction is closely related to the topological properties of ancestor TCs. In particular, we distinguish ancestor TCs into two types, by introducing the following concepts:

Definition 9. *(**Bad parent TC**:) For a non-root TC $\mathcal{T}$, if its parent TC is a 3-vertex-connected TC $\mathcal{T}_{3vc}$ with 2-bridge-cut link in $\widetilde{\mathcal{T}}_{3vc}$ (where $\widetilde{\mathcal{T}}_{3vc}$ is the subgraph of $\mathcal{T}_{3vc}$ by removing the direct link between $\mathcal{T}_{3vc}$'s two vantage points) or shares vantages with its children (no matter with $\mathcal{T}$ or $\mathcal{T}$'s siblings), we call the parent TC a bad parent, otherwise a good parent.*

As an example, in Fig. 6.8, $\mathcal{T}_2$ is a bad parent of $\mathcal{T}_3$, because $\mathcal{T}_2$ is a $\mathcal{T}_{3vc}$ with 2-bridge-cut links (l_{v_4,v_5}, l_{v_8,v_9}), but in Fig. 6.9, there is no bad parent.

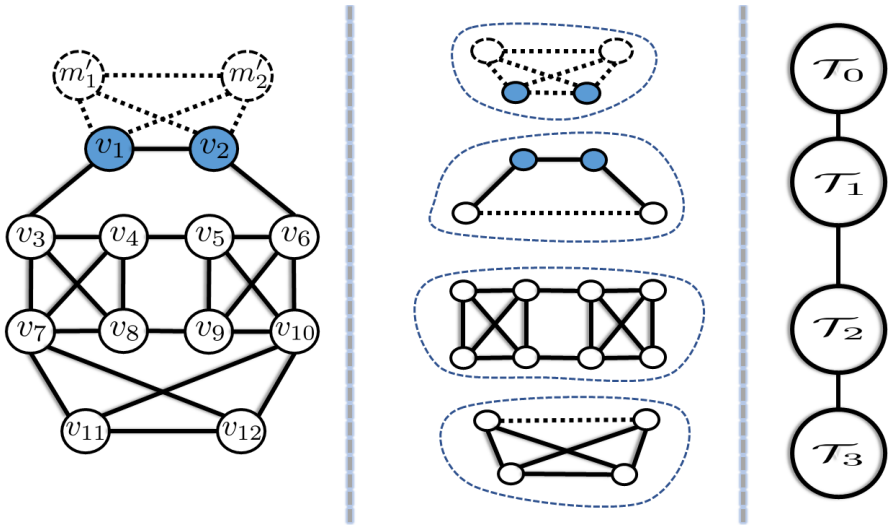


Figure 6.8: $\mathcal{T}_2$ is a bad parent of $\mathcal{T}_3$.

The main difference between a bad parent and a good parent is that the good parent does not leave much room for us to use SLM. This is because with a good parent, the metric of $S_{m_1,\mu_1}+S_{\mu_2,m_2}$ (shown in Fig. 6.7) can be determined or there is only one possible segment combination in this good parent TC to comprise $S_{m_1,\mu_1}+S_{\mu_2,m_2}$ (refer to [19,31] for identifiability analysis). In this case, we only need to focus on building the path segment inside $\mathcal{T}$, and thus there is not enough room for taking advantage of SLM. If the parent TC is bad, however, the metric of $S_{m_1,\mu_1}+S_{\mu_2,m_2}$ cannot be determined as it has multiple possible combinations (refer to [19, 31] for identifiability analysis), and in these cases if we first ignore $\mathcal{T}$ and focus on measuring and collecting data to determine the *smallest* value of $S_{m_1,\mu_1}+S_{\mu_2,m_2}$, then we can reduce the possible path combinations when we move

down to considering unidentifiable links in $\mathcal{T}$. This is because path segments that lead to *minimum* $S_{m_1,\mu_1} + S_{\mu_2,m_2}$ have already been recorded in previous measurement data, and thus we only need to consider those path segments when we move down to $\mathcal{T}$. Clearly, this strategy is not possible for OBM.

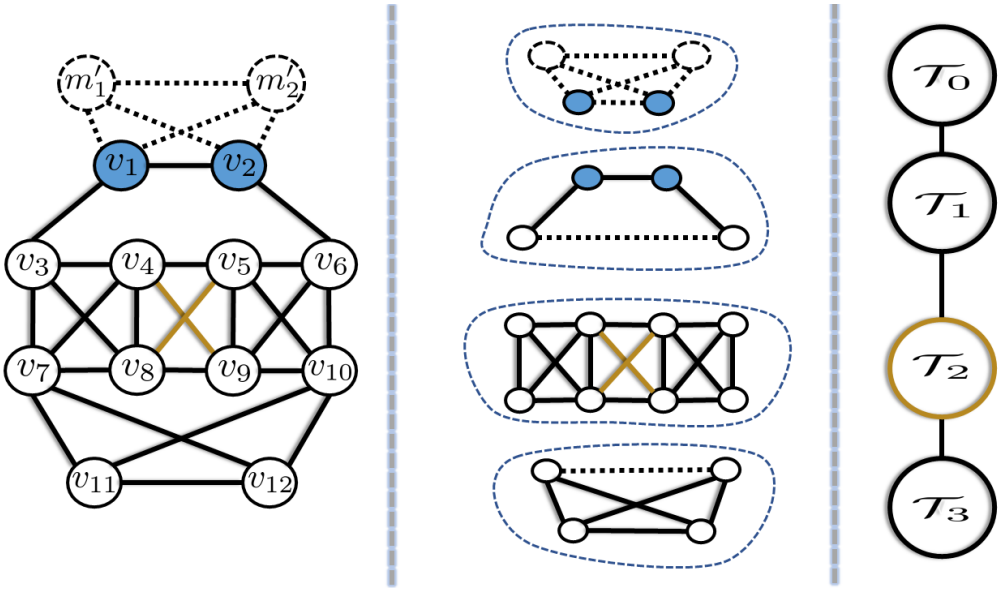


Figure 6.9: $\mathcal{T}_2$ is a good parent of $\mathcal{T}_3$.

6.3.3 Details of SLM

To ease explanation, we use $\mathcal{MP}_i$ to denote MPs specially targeted for the unidentifiable links in TC $\mathcal{T}_i$.

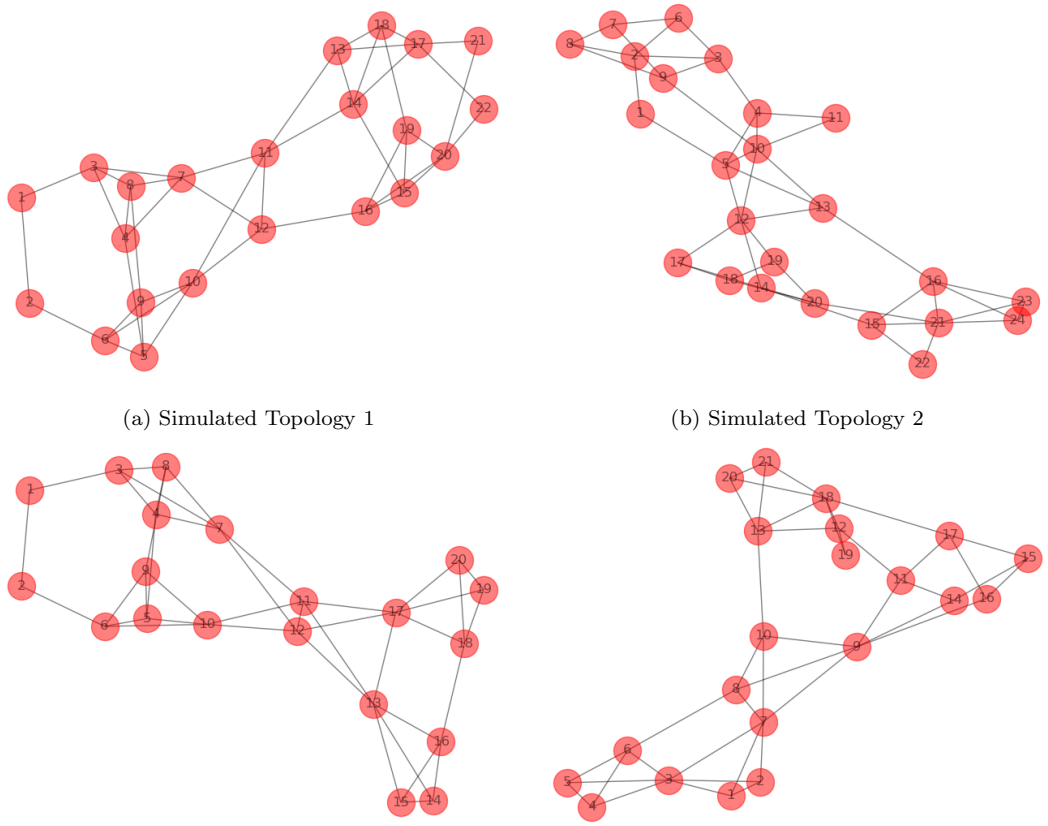
Simply put, SLM constructs MPs layer by layer following the hierarchy of the TC-tree: it first builds MPs for $\mathcal{T}_0$'s children, denoted as $\mathcal{T}_1$ layer, and collects measurement data; by looking at the measurement data for $\mathcal{T}_1$ layer, it excludes some unneeded MPs and then constructs the MPs for the grandchild TCs of $\mathcal{T}_0$. The above process is repeated until all TCs are covered.

We now use the example in Fig. 6.8 to explain the details of SLM, in comparison with the OBM method such as the minimal measurement path construction (MMPC) Algorithm [19]. First, we build $\mathcal{MP}_1$: $\mathcal{T}_1$ is a circle TC, in which every link is unidentifiable except the direct link between the two monitors (i.e., l_{v_1,v_2} is identifiable, but l_{v_1,v_3}, l_{v_3,v_4}, l_{v_4,v_5}, l_{v_5,v_6}, l_{v_2,v_6} are all unidentifiable). Hence we can construct one and only one MP to cover all the links in $\mathcal{T}_1$, i.e., $v_1-v_3-v_4-v_5-v_6-v_2$. We collect the measurement along this path.

Then we move down to the next level and build $\mathcal{MP}_2$. Regardless of the segment of MPs within $\mathcal{T}_2$, any $\mathcal{MP}_2$ will traverse through and back to $\mathcal{T}_1$. Note that v_3 and v_6 are the vantage nodes w.r.t. $\mathcal{T}_2$. There is only 1 possible segment combination, i.e., $l_{v_1,v_3}+l_{v_2,v_6}$, for MPs passing through $\mathcal{T}_1$ and covering $\mathcal{T}_2$. We can construct 18 MPs that concatenate with l_{v_1,v_3} and l_{v_2,v_6}: since there are three exterior links (unidentifiable) incident to vantage v_3, two 2-bridge-cut links (unidentifiable), and three exterior links incident to the other vantage v_6, we need to build all the possible combinations

of them ($3 \times 2 \times 3 = 18$) such that the tightest error bound is guaranteed. We then go ahead to collect measurement data along all the 18 paths.

We continue to build $\mathcal{MP}_3$. The benefit of sequential measurement will appear, because $\mathcal{T}_2$ is a bad parent of $\mathcal{T}_3$. Same as before, path $\mathcal{MP}_3$ will traverse through and back to $\mathcal{T}_2$. The total number of path segments in $\mathcal{T}_2$ that need to include in $\mathcal{MP}_3$ is $3 \times 3 = 9$ (three exterior link incident to v_3 and three exterior link incident to v_6). If we use SLM, among the 9 possible combinations, we *only* need to include the one(s) that have the smallest metric sum, with one exterior link incident to v_3 and one exterior link incident to v_6. This information has already been stored in the measurement results from $\mathcal{MP}_2$. In contrast, OBM does not collect probing measurements in $\mathcal{MP}_2$ and thus has no knowledge[1] of which route has the smallest metric sum in $\mathcal{T}_2$. As a consequence, OBM must build MPs that list all possible combinations.



(a) Simulated Topology 1

(b) Simulated Topology 2

(c) Simulated Topology 3

(d) Simulated Topology 4

Figure 6.10: Simulated networks for comparing SLM and OBM.

6.3.4 The Benefit of SLM over OBM

As shown in Section 6.3, SLM can reduce the number of MPs whenever there are bad parent TCs in the TC tree. The real-world ISP networks include mainly backbone routers and their topology may not be diverse. To illustrate the benefit of SLM, we need diverse networks consisting of a number of

[1]The information cannot be inferred because $\mathcal{T}_2$ is a bad parent, in which the sum of two exterior links is unknown.

bad parents TCs. As such, we simulate four networks, whose details are illustrated in Fig. 6.10. For each network, we run tests 100 times, and in each test we randomly select 10% nodes as monitoring nodes and compare the number of MPs obtained with SLM and that with OBM.

The results are shown in Fig. 6.11. From the figure, we can clearly see the benefit of SLM over OBM. *In every single test*, SLM returns a smaller number of MPs than that with OBM. *On average*, SLM can reduce the number of MPs with OBM, **ranging from** 18% **to** 53% **reduction** in the tested scenarios. In addition, the results from SLM have much smaller variations.

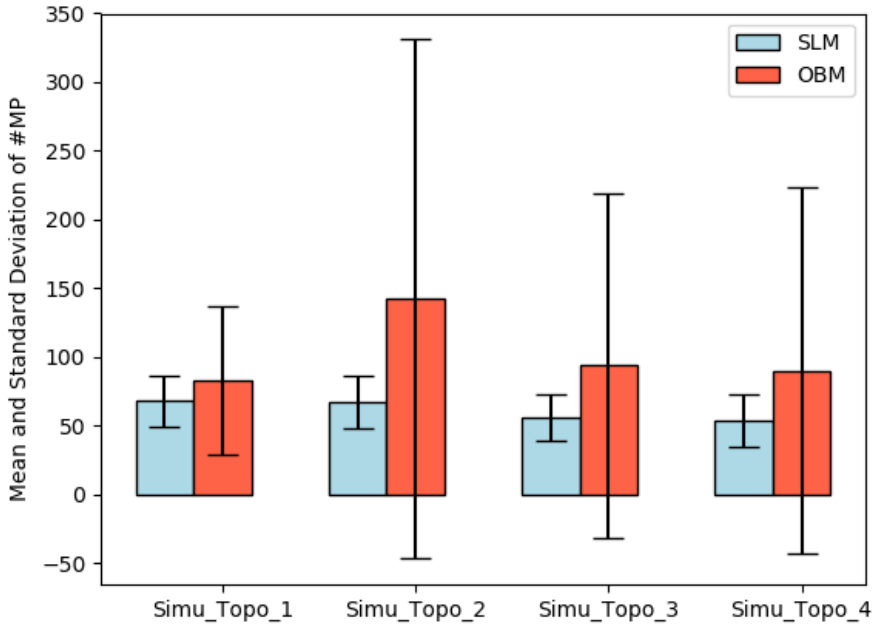


Figure 6.11: Benefit of SLM over OBM

Chapter 7

Conclusions and Future Work

7.1 Conclusion

In this thesis, we initiated two network tomography researches: concentrated network tomography and bound-based network tomography. Both are motivated by practical needs in that network operators normally care more about the performance of critical paths (or interested paths) and the performance bounds whenever exact performance values cannot be determined.

In the context of concentrated network tomography, we studied the problem of inferring the metrics of a set of given interested paths. In particular, we disclosed the necessary and sufficient condition for the identifiability of a given path, and developed an efficient algorithm that uses the minimum number of monitors, whose measurements can be used to identify the performance of interested paths. Experimental results show that compared to other link-based solutions, our solution leads to a saving of up to 40% fewer monitors. This work is practically relevant to the current trend of "network-as-service", and will have more emerging applications such as verifiable network services. It makes new theoretical contributions to the area of network tomography, in the sense that concentrated network tomography a new and challenging variant of network tomography.

In the context of bound-based network tomography, we extended Boolean-based network tomography to bound-based network tomography. To the best of our knowledge, this is the first work that contributes a whole new study field in network tomography, in which not only existing results on link identifiability in Boolean-based network tomography hold as before, but also performance bounds on unidentifiable links are derived. While conceptually simple, bound-based network tomography poses much harder technical challenges and renders existing Boolean-based solutions suboptimal or even infeasible. In this work, we tackled several core technical challenges in bound-based network tomography, including how to derive the tightest total error bound, how to build least measurement paths for deriving the tightest performance bounds, and how to deploy extra monitors to tighten performance bounds over existing ones. Bound-based network tomography has significant practical implication, because ISPs can effectively diagnose network problems and make better decisions with the rich information on performance bounds.

In bound-based network tomography, we also studied the problem of controlling the maximum error bound over all network links. We developed a method that can be used repeatedly to push down

the maximum link error bound. The method is based on graph decomposition and has theoretical guarantees on the minimum number of new monitors required to reduce the maximum link error bound. In addition, it can tell the best places for the new monitor deployment. We also proposed a sequential learning-based measurement (SLM) method to construct measurement paths, which can reduce the number of measurement paths required to obtain the tightest link error bound.

Overall, the theoretical results and the algorithms presented in this thesis are directly applicable to network performance management in various types of networks, where directly measuring all links is practically impossible.

7.2 Future Work

Along the line of concentrated network tomography and bound-based network tomography, there are many interesting (open) research problems:

- In many cases, the link performance is asymmetric. For example, the delay of a link on one direction may be different from that on the other direction. In this case, we have to use directed graph to model the networks. The theoretical analysis in this thesis does not hold anymore. Extending our work to directed graphs is a challenging research problem.

- While we have studied optimal measurement path construction methods in bound-based network tomography, measurement path construction, on its own merit, deserves further investigation. Our optimal measurement path construction methods are based on some assumptions, e.g., we are not allowed to use trail-and-error, and we used the mean value of the lower and upper bounds on each link to estimate the delay of a path between an existing monitor and a candidate node (if the new monitor be deployed at the candidate node). Relaxing these assumptions may open new opportunities to use reinforcement learning in designing intelligent methods for measurement path construction.

- In this thesis, we do not consider any malicious behavior of intermediate routers. Reliable network tomography will be a very interesting research if we assume that some intermediate routers may inject fake information.

Appendix A

List of Publications from the Thesis

- R.W. Yang, **C.Y. Feng**, L.N. Wang, W. Wu, K. Wu, J.P. Wang, Y.L. Xu, "On the Optimal Monitor Placement for Inferring Additive Metrics of Interested Paths," In the Proceedings of *IEEE International Conference on Computer Communications (Infocom)*, Honolulu, HI, April. 2018. [acceptance rate: 19.7%]
- **C.Y. Feng**, L.N. Wang, K. Wu, J.P. Wang, "Bound-based Network Tomography with Additive Metrics," In the Proceedings of *IEEE International Conference on Computer Communications (Infocom)*, Paris, France, April. 2019. [acceptance rate: 19.2%]
- **C.Y. Feng**, L.N. Wang, K. Wu, J.P. Wang, "Bound Inference in Network Performance Tomography with Additive Metrics," *ACM/IEEE Transactions on Networking*, accepted in June 2020.

Bibliography

[1] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck. Network slicing and softwarization: A survey on principles, enabling technologies, and solutions. *IEEE Communications Surveys & Tutorials*, 20(3):2429–2453, 2018.

[2] S. S. Ahuja, S. Ramasubramanian, and M. Krunz. Srlg failure localization in all-optical networks using monitoring cycles and paths. In *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, pages 700–708, 2008.

[3] Y. Bejerano and Rajeev Rastogi. Robust monitoring of link delays and faults in ip networks. In *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, volume 1, pages 134–144 vol.1, 2003.

[4] Y. Bejerano and R. Rastogi. Robust Monitoring of Link Delays and Faults in IP Networks. *IEEE/ACM Transactions on Networking*, 14(5):1092–1103, 2006.

[5] R. Caceres, N. G. Duffield, J. Horowitz, and D. F. Towsley. Multicast-based inference of network-internal loss characteristics. *IEEE Transactions on Information Theory*, 45(7):2462–2480, 1999.

[6] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu. Network tomography: Recent developments. *Statistical Science*, 19(3):499–517, 2004.

[7] R. M. Castro, M. J. Coates, and R. D. Nowak. Likelihood based hierarchical clustering. *IEEE Transactions on Signal Processing*, 52(8):2308–2321, 2004.

[8] A. Chen, J. Cao, and T. Bu. Network tomography: Identifiability and fourier domain estimation. *IEEE Transactions on Signal Processing*, 58(12):6029–6039, 2010.

[9] Y. Chen, D. Bindel, H. Song, and R. H. Katz. An Algebraic Approach to Practical and Scalable Overlay Network Monitoring. In *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 55–66, New York, NY, USA, 2004. ACM.

[10] W. Cheney and D. R. Kincaid. *Linear Algebra: Theory and Applications*. Jones and Bartlett Publishers, Inc., 1st edition, 2008.

[11] A. Coates, A. O. Hero III, R. Nowak, and Bin Yu. Internet tomography. *IEEE Signal Processing Magazine*, 19(3):47–65, 2002.

[12] B. K. Dey, D. Manjunath, and S. Chakraborty. Estimating network link characteristics using packet-pair dispersion: A discrete-time queueing theoretic analysis. *Computer Networks*, 55(5):1052 – 1068, 2011.

[13] G. Di Battista and R. Tamassia. On-line planarity testing. *SIAM Journal on Computing*, 25(5):956–997, 1996.

[14] 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 Transactions on Networking*, 25(1):210–223, 2017.

[15] A. B. Downey. Using pathchar to estimate internet link characteristics. *SIGCOMM Comput. Commun. Rev.*, 29(4):241–250, Aug. 1999.

[16] N. Duffield, F. Lo Presti, V. Paxson, and D. Towsley. Network loss tomography using striped unicast probes. *IEEE/ACM Transactions on Networking*, 14(4):697–710, 2006.

[17] B. Eriksson, G. Dasarathy, P. Barford, and R. Nowak. Efficient network tomography for internet topology discovery. *IEEE/ACM Transactions on Networking*, 20(3):931–943, 2012.

[18] N. Etemadi Rad, Y. Ephraim, and B. L. Mark. Delay network tomography using a partially observable bivariate markov chain. *IEEE/ACM Transactions on Networking*, 25(1):126–138, 2017.

[19] C. Feng, L. Wang, K. Wu, and J. Wang. Bound-based network tomography with additive metrics. In *IEEE INFOCOM*, Paris, France, April 2019.

[20] 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 Transactions on Networking*, 24(3):1392–1403, 2016.

[21] A. Gopalan and S. Ramasubramanian. On Identifying Additive Link Metrics Using Linearly Independent Cycles and Paths. *IEEE/ACM Transactions on Networking*, 20(3):906–916, 2012.

[22] A. Gopalan and S. Ramasubramanian. On the maximum number of linearly independent cycles and paths in a network. *IEEE/ACM Transactions on Networking*, 22(5):1373–1388, 2014.

[23] O. Gurewitz and M. Sidi. Estimating one-way delays from cyclic-path delay measurements. In *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213)*, volume 2, pages 1038–1044 vol.2, 2001.

[24] C. Gutwenger and P. Mutzel. A linear time implementation of spqr-trees. In *International Symposium on Graph Drawing*, pages 77–90. Springer, 2000.

[25] J. D. Horton and A. López-Ortiz. On the Number of Distributed Measurement Points for Network Tomography. In *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, pages 204–209, New York, NY, USA, 2003. ACM.

[26] N. Hu, L. E. Li, Z. M. Mao, P. Steenkiste, and J. Wang. Locating internet bottlenecks: Algorithms, measurements, and implications. *SIGCOMM Comput. Commun. Rev.*, 34(4):41–54, Aug. 2004.

[27] R. Kumar and J. Kaur. Practical Beacon Placement for Link Monitoring Using Network Tomography. *IEEE Journal on Selected Areas in Communications*, 24(12):2196–2209, 2006.

[28] L. Ma, T. He, K. K. Leung, A.Swami and D.Towsley. Identifiability of Link Metrics Based on End-to-end Path Measurements. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, pages 391–404, New York, NY, USA, 2013. ACM.

[29] E. Lawrence, G. Michailidis, V. Nair, and B. Xi. Network tomography: A review and recent developments. *Ann Arbor*, 1001(48):109–1107, 2006.

[30] F. Lo Presti, N. G. Duffield, J. Horowitz, and D. Towsley. Multicast-based inference of network-internal delay distributions. *IEEE/ACM Transactions on Networking*, 10(6):761–775, 2002.

[31] 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 Transactions on Networking (TON)*, 22(4):1351–1368, 2014.

[32] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley. Monitor placement for maximal identifiability in network tomography. In *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, pages 1447–1455, 2014.

[33] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley. Partial network identifiability: Theorem proof and evaluation. Technical report, Imperial College, London, UK, 2014.

[34] L. Ma, T. He, K. K. Leung, D. Towsley, and A. Swami. Efficient Identification of Additive Link Metrics via Network Tomography. In *2013 IEEE 33rd International Conference on Distributed Computing Systems*, pages 581–590, 2013.

[35] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. Inferring link weights using end-to-end measurements. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment*, IMW '02, page 231–236, New York, NY, USA, 2002. Association for Computing Machinery.

[36] M. Mardani and G. B. Giannakis. Estimating traffic and anomaly maps via network tomography. *IEEE/ACM Transactions on Networking*, 24(3):1533–1547, 2016.

[37] J. Ni, H. Xie, S. Tatikonda, and Y. R. Yang. Efficient and dynamic routing topology inference from end-to-end measurements. *IEEE/ACM Transactions on Networking*, 18(1):123–135, 2010.

[38] V. N. Padmanabhan, L. Qiu, and H. J. Wang. Server-based inference of internet link lossiness. In *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, volume 1, pages 145–155 vol.1, 2003.

[39] F. L. Presti, N. G. Duffield, J. Horowitz, and D. Towsley. Multicast-based inference of network-internal delay distributions. *IEEE/ACM Transactions on Networking*, 10(6):761–775, 2002.

[40] N. Spring, R. Mahajan, D. Wetherall, and H. Hagerstrom. Rocketfuel: An ISP topology mapping engine, 2002.

[41] R. Tarjan. Depth-First Search and Linear Graph Algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972.

[42] S. Tati, S. Silvestri, T. He, and T. L. Porta. Robust network tomography in the presence of failures. In *2014 IEEE 34th International Conference on Distributed Computing Systems*, pages 481–492, 2014.

[43] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

[44] Y. Vardi. Network Tomography: Estimating Source-Destination Traffic Intensities from Link Data. *Journal of the American Statistical Association*, 91(433):365–377, 1996.

[45] Y. Xia and D. Tse. Inference of Link Delay in Communication Networks. *IEEE Journal on Selected Areas in Communications*, 24(12):2235–2248, 2006.

[46] R. Yang, C. Feng, L. Wang, W. Wu, K. Wu, J. Wang, and Y. Xu. On the optimal monitor placement for inferring additive metrics of interested paths. In *IEEE INFOCOM*, Honolulu, HI, April 2018.

[47] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg. Fast accurate computation of large-scale ip traffic matrices from link loads. In *Proceedings of the 2003 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '03, page 206–217, New York, NY, USA, 2003. Association for Computing Machinery.

[48] E. Zhao and L. Tan. A pca based optimization approach for ip traffic matrix estimation. *Journal of Network and Computer Applications*, 57:12 – 20, 2015.

[49] Y. Zhao, Y. Chen, and D. Bindel. Towards unbiased end-to-end network diagnosis. *SIGCOMM Comput. Commun. Rev.*, 36(4):219–230, Aug. 2006.

[50] Q. Zheng and G. Cao. Minimizing probing cost and achieving identifiability in probe-based network link monitoring. *IEEE Transactions on Computers*, 62(3):510–523, 2013.