Bandwidth Tomography

by

Jianwei An
B.Sc., Wuhan University, 2017

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

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

We acknowledge and respect the Lək̓ʷəŋən (Songhees and Xʷsepsəm/Esquimalt) Peoples on whose territory the university stands, and the Lək̓ʷəŋən and W̱SÁNEĆ Peoples whose historical relationships with the land continue to this day.

Bandwidth Tomography

by

Jianwei An
B.Sc., Wuhan University, 2017

Supervisory Committee

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

Dr. Jianping Pan, Departmental Member
(Department of Computer Science)

Dr. Xiaodai Dong, Outside Member
(Department of Electrical and Computer Engineering)

ABSTRACT

Bandwidth tomography—inferring the bandwidth of internal network links from end-to-end path bandwidth measurements—is a long-standing open problem in network tomography. The core challenge arises from the fact that no existing mathematical framework directly addresses the inverse problem formulated as a set of *min*-equations.

To systematically tackle this challenge, we design a polynomial-time algorithm that accurately determines the bandwidth of all identifiable links and derives the tightest possible error bounds for unidentifiable links based on a given set of measurement paths. Furthermore, when additional information on link correlations is available, we leverage the extra information to refine our error bounds. Specifically, we explore two key types of link correlations: fairness constraints and total capacity constraints among a node's adjacent links. We provide theoretical guarantees on how these correlations enhance the precision of bandwidth tomography and develop algorithms to address two fundamental challenges in refining these bounds: (i) the impact of synchronous vs. asynchronous updates and (ii) the cascading effects during bound updates.

Having developed algorithms to derive the tightest possible performance bounds for a given set of measurement paths, we then tackle the next major challenge: constructing optimal measurement paths that minimize the global error bounds for unidentifiable links. We prove the hardness of this problem and, in response, propose a reinforcement learning (RL) approach for measurement path construction. Our solution leverages domain-specific knowledge in bandwidth tomography and integrates both offline training and online prediction to build suitable measurement paths.

We evaluate our proposed methods using real-world ISP topologies and simulated networks. Experimental results show that compared to existing path construction methods—Random and Diversity Preferred—our RL-based approach significantly reduces the average error bound of inferred link bandwidths. In addition, our performance bound computation algorithms improve the state-of-the-art techniques by substantially tightening the performance bounds in bandwidth tomography.

Contents

Supervisory Committee **ii**

Abstract **iii**

Table of Contents **iv**

List of Tables **vi**

List of Figures **vii**

Acknowledgements **viii**

Dedication **ix**

1 Introduction **1**

1.1 Background: Network Tomography 1

1.1.1 Performance Tomography 1

1.1.2 Structural Tomography 2

1.1.3 Fault and Anomaly Detection 2

1.1.4 Traffic Matrix Estimation 3

1.1.5 Recent Advances: Machine Learning-Based Methods 3

1.2 Why Is Bandwidth Tomography Hard? 4

1.3 Practical Significance of Bandwidth Tomography 5

1.4 Research Contributions 6

2 The Basic Model and Solution **9**

2.1 The Basic System Model 9

2.2 Bound Analysis and Algorithm for Obtaining the Tightest Error Bound 12

3 The Constrained Model and Solution **19**

3.1 Motivation 19
3.2 The Constrained System Model 20
3.3 Bound Analysis and Algorithm for Obtaining the Tightest Error Bound 24
3.3.1 Elaboration on the Tightest Error Bound 24
3.3.2 Challenges in Obtaining the Tightest Error Bounds 26
3.4 Tackling Challenge 1: The CBLN algorithm and the Local Tightest Error Bounds 27
3.5 Moving from Local to Global Optimality 31
3.6 Tackling Challenge 2: The GES algorithm and The Global Tightest Error Bounds 32
3.7 Conclusion 34

4 A Reinforcement Learning Approach for Path Construction 39
4.1 On the Hardness of Path Construction for the (Global) Tightest Error Bound 39
4.2 Special Knowledge in Bandwidth Tomography for Action Design . . . 44
4.3 Guided Sequential Path Construction (GSPC) 46
4.3.1 Overview 46
4.3.2 Policy 48

5 Performance Evaluation 51
5.1 Evaluation of GSPC 51
5.1.1 Experiment on Real-World ISP Networks 52
5.1.2 Performance of *GSPC* v.s. Theoretical Smallest TEB (TS-TEB) 58
5.1.3 The Stability of GSPC 59
5.2 Evaluation of the GES algorithm 60
5.2.1 Performance Results 62
5.2.2 The Impact of ϵ 62
5.3 Summary of Evaluation Results 66

6 Conclusions and Future Work 67
6.1 Conclusions 67
6.2 Future Work 68

Bibliography 70

List of Tables

Table 2.1 Dividing links into disjoint sets 16

Table 3.1 Main Notations Used in This Chapter 25

Table 5.1 Network parameters 54
Table 5.2 Ratio of the number of identified links over the total number of links 54
Table 5.3 Average GSPC Performance 60
Table 5.4 Network Parameters 61
Table 5.5 Performance Comparison between GES and CTB 62

List of Figures

Figure 2.1 Example topology (a) and its simplified topology (b). 10
Figure 2.2 An example network to illustrate CTB. 14

Figure 3.1 An example network with correlated links caused by load balancing. v_1, v_2, and v_3 are also monitoring nodes. 23

Figure 4.1 Illustration of *I-structure*: P_1 and P_2 are marked in red and cyan, respectively; every link of P_1 except $l_{2,4}$ is also a link of P_2. . . . 45

Figure 5.1 Performance of random, diversity preferred and GSPC. 52
Figure 5.2 The distribution of error bounds. 53
Figure 5.3 An example small network where we can list all possible MPs (2 monitors are in red color). 55
Figure 5.4 Compare the performance of *GSPC* with theoretical smallest TEB (TS-TEB). 56
Figure 5.5 Stability experiment: (a) (b) results with different ground-truth sets, and (c) (d) results with different monitor sets. 57
Figure 5.6 The performance of GES over different update sequences with different ϵ values. 63
Figure 5.7 The average fluctuation under different ϵ values over 100 GES runs. 64
Figure 5.8 The ratio of the average times of links' extra updates under different ϵ values over 100 GES runs. 65

ACKNOWLEDGEMENTS

I would like to thank:

Supervisor, Dr. Kui Wu, for his patience and for supporting me in the low moments.

Committee members, Dr. Jianping Pan and Dr. Xiaodai Dong, for providing me with insightful feedback on my disseration.

External member, Dr. Yuanzhu Chen, for his kindness in serving my oral examination.

I believe I know the only cure, which is to make one's centre of life inside of one's self, not selfishly or excludingly, but with a kind of unassailable serenity-to decorate one's inner house so richly that one is content there, glad to welcome any one who wants to come and stay, but happy all the same in the hours when one is inevitably alone.

Edith Wharton

DEDICATION

To my mom.

Chapter 1

Introduction

1.1 Background: Network Tomography

To guarantee the quality of service (QoS) and the smooth operation of networks, any Internet service provider (ISP) needs to closely monitor the network performance, including delay, available bandwidth, network congestion, link losses, and so on. In a large physical network, directly measuring the performance of each link is infeasible due to the high measurement overhead. In the new era of network virtualization, ISPs can utilize network slices to dynamically form different virtual networks for dedicated network applications. Network virtualization, however, does not necessarily lead to easy performance monitoring. In contrast, due to the dynamic changes of network configuration, it becomes more critical yet more challenging to directly measure the performance of virtual links or virtual services. A well-known strategy is to infer the performance/status of physical/virtual links via end-to-end measurements. This method was first termed as network tomography [1] in 1996 and has attracted substantial research since then [2–13]. Since direct observations of internal network states are often infeasible due to privacy concerns or infrastructure limitations, network tomography provides a powerful approach for performance monitoring, fault diagnosis, and capacity planning. Over the years, research in network tomography has evolved into several distinct categories, each addressing different aspects of network inference [14].

1.1.1 Performance Tomography

This category focuses on estimating performance metrics such as latency, packet loss, and available bandwidth by analyzing end-to-end path measurements. The primary

challenge in performance tomography lies in solving underdetermined inverse problems where internal network conditions must be inferred from aggregated observations. Depending on the performance metrics of interest, performance tomography includes:

- Delay tomography: Infers per-link delay using packet probes and statistical inference models [2, 3, 14].
- Loss tomography: Estimates link loss rates through correlation-based inference and maximum likelihood estimation [14].
- Bandwidth tomography: Determines the bandwidth of internal links by solving inverse problems involving *min*-equations (i.e., bandwidth bottlenecks). This thesis falls within this special type of performance tomography.

It is worth mentioning that delay tomography and loss tomography are mathematically the same because both involve additive metrics, meaning that the path delay is equal to the delay of individual links along the path[1]. Nevertheless, the performance metric in bandwidth tomography is not additive.

1.1.2 Structural Tomography

Structural tomography aims to reconstruct network topology based on observed path measurements. Unlike performance tomography, which infers link-level characteristics, structural tomography seeks to determine network connectivity and routing structures. The main approaches include:

- Binary tomography: Uses multicast or unicast probes to infer whether a given link exists [15–17].
- Topology discovery: Applies graph-based inference and machine learning to reconstruct network structures from path traces [18, 19].

1.1.3 Fault and Anomaly Detection

This line of research detects and localizes network failures, congestion, or malicious activities. Methods often involve [14]:

[1]Path loss rate by itself is not the sum of link loss rates, but after taking the logarithm operation on loss rate, we can translate loss rate to additive metric.

- Passive inference: Identifying anomalies by analyzing passively collected traffic data.
- Active probing: Deploying probe packets to pinpoint failure locations and estimate their impact.

1.1.4 Traffic Matrix Estimation

Traffic matrix estimation focuses on inferring the volume of traffic flow between network nodes using measurements of load on carefully chosen links. Since direct traffic flow data is often unavailable, statistical inference and optimization techniques are used to approximate the underlying traffic distribution. Existing methodologies can be roughly grouped into:

- Linear inverse modeling: Uses aggregated link loads to estimate traffic matrices [14].
- Bayesian inference and machine learning: Enhances accuracy by integrating prior knowledge and adaptive learning methods [20].

1.1.5 Recent Advances: Machine Learning-Based Methods

Recent advances in network tomography have increasingly incorporated deep machine learning, probabilistic modeling, and reinforcement learning to improve inference accuracy and scalability. Notable developments include:

- Deep learning for network inference: Utilizing neural networks to model complex network behaviors [21–23].
- Graph-based methods: Applying graph neural networks (GNNs) for network state estimation [24].
- Reinforcement learning for path selection: Optimizing measurement path construction using RL to improve inference precision [25].

Overall, network tomography remains a critical research area with applications in network monitoring, security, and optimization. While traditional statistical and optimization-based methods have provided foundational insights, emerging AI-driven approaches continue to push the boundaries of inference accuracy and efficiency.

Within this large research context, this thesis focuses on tackling the unique challenges of bandwidth tomography.

1.2 Why Is Bandwidth Tomography Hard?

So far, most work on network tomography focuses on additive metrics, e.g., delay, where the value of an end-to-end path is the total value of all the links on the path. An important performance metric, available bandwidth (bandwidth for short[2]), has been largely avoided in the network tomography literature. It is well known that many research papers have investigated the method to estimate the end-to-end bandwidth [26], and accordingly measurement tools have been developed. For instance, *pathchar*, *clink*, *pchar*, and *bfind* use ICMP "TTL exceeded" messages for bandwidth estimation. Nevertheless, all the above research mainly targets at estimating the end-to-end bandwidth. While some tools such as *Tailgating* [27] and *pathneck* [28] can use probe packets to estimate the bandwidth of a link along the measurement path. These tools are inaccurate on paths longer than a few hops [27] or require intermediate routers to support ICMP [28]. Actually, assuming the capability of estimating the metric (e.g., bandwidth/delay) of links along a path just from the end-to-end measurement of this path would invalidate most, if not all, work in network tomography. Overall, a network tomography approach to estimating the bandwidth of every *individual link* in the whole network via a small number of monitoring nodes, termed as *bandwidth tomography*, is surprisingly missing.

Boolean-based network tomography that identifies link failure or link congestion is loosely relevant but starkly different from bandwidth tomography. The goal of Boolean-based network tomography [29] is to infer whether or not a link/node is congested or failed based on the congestion/failure status of end-to-end paths. In other words, the input and output of Boolean-based network tomography are both binary. In contrast, bandwidth tomography is to infer the bandwidth *value* of individual links from end-to-end path bandwidth.

A piece of work in the network calculus domain [30, 31] has touched a similar problem as bandwidth tomography: inferring the service curves of links based on

[2]The term bandwidth is overused in the literature. It may refer to either the *maximum* transfer capacity of a link (i.e., the link capacity), or the *actual* amount of data that can be sent and received within a specific time frame (i.e., available bandwidth or throughput). The term refers to the latter case in this thesis.

the service curve of end-to-end path, i.e., service curve decomposition. Service curve denotes the *cumulative* service amount *over time* offered to a given traffic flow. Nevertheless, only line topology and tree topology were studied. The limited results in [30], the fact that no significant progress has been found since 2008, and our own investigation, all suggest that service curve decomposition is a hard-to-achieve goal in general networks.

The bandwidth tomography research is missing in the literature for two main reasons.

- First, unlike additive metrics, bandwidth is a metric that uses the minimum, i.e. the bandwidth of a path equals the minimal bandwidth over all the links along the path. No existing mathematical tool can be applied directly to solve the inverse problem with a set of *min*-equations (refer to Chapter 2 for details). The only analytical tool that might help is max-plus [32] and min-plus [33] algebras. Our deep investigation, however, concludes that these algebras cannot be used to solve our problem since the *min*-equations do not satisfy the algebraic properties of max-plus [32] and min-plus [33].

- Second, the minimum operation results in high information loss, since we only know that the bandwidth of a path is not higher than the bandwidth of the constituting links. As an analogy to lossy compression, the minimum operation on link bandwidth values is similar to a quantizer that replaces all the values with the same (smallest) value. Without the help of other side information, it is theoretically impossible to recover the original values after this quantization step.

1.3 Practical Significance of Bandwidth Tomography

As a special type of network tomography, bandwidth tomography poses unique challenges but has profound practical implications. While network tomography has not been used yet as a *standalone* technology on the Internet, it can work together with existing tools and broaden their functionality. For instance, using the popular network measurement tools such as *pathchar*, *clink*, and *pchar* as the basic building block for measuring the end-to-end bandwidth, our algorithms (1) guide the construction

of measurement paths and (2) return the network-wide, link-level bandwidth results. In-band network telemetry (INT) [34], which is arguably the most advanced network monitoring technique, allows individual devices to report the statistics in the data plane directly to monitoring applications running in the centralized SDN control plane. INT, if deployed and enabled in every device, would void the practical need for network tomography as a whole. Nevertheless, large-scale INT is faced with non-trivial technical challenges such as orchestration, data aggregation, security, and high monitoring overhead [34]. At least in the near future, INT must work with legacy devices and cannot be deployed at large scale. We believe that INT can benefit from network tomography to tackle the above challenges.

Interestingly, the idea of bandwidth tomography has recently found applications in other domains, such as in the cryptocurrency domain. The Lightning Network [35] is an advanced technology that improves the scalability of Bitcoin and other cryptocurrencies. Transactions on the Lightning Network are facilitated through a system of channels. When two parties wish to transact, they open a payment channel between them, allowing for an unlimited number of transactions. These transactions are not individually recorded on the Bitcoin blockchain; instead, only the final state of the channel is recorded when the channel is closed. The network also supports the routing of payments across multiple channels, enabling users to send payments to each other even if they do not have a direct channel open. To support this, it is required that all balances along the payment path be higher than the transaction amount. In other words, we can formulate a *min* equation along the payment path. An interesting question is that the account balances are hidden from the public for privacy and security reasons. It has been shown that methods from bandwidth tomography can be utilized to infer the account balances in the Lightning Network efficiently [36].

1.4 Research Contributions

We, for the first time, formally formulate and systematically study three core problems in bandwidth tomography:

1. Given a network, a set of end-to-end measurement paths, and the measured end-to-end bandwidth, is a link in the network identifiable[3]? If not, what are the lower and upper bounds of its bandwidth?

[3]When the value of a link can be uniquely determined, we call the link *identifiable* [37].

2. If extra information, e.g., the fairness constraint and the total capacity constraint for links adjacent to a common node, is available, can we leverage it to refine the performance bounds of bandwidth tomography?

3. How to construct measurement paths so that any link is (a) either identifiable or (b) the error bound (i.e., the gap between the upper and lower bounds) is the smallest if its identifiability is impossible.

This thesis answers the above challenging questions and lays a solid foundation for future research in bandwidth tomography. The contributions of the thesis include:

- For the first problem, we develop a polynomial-time algorithm that returns the exact bandwidth value for all identifiable links and the smallest error bounds for unidentifiable links. We also present the necessary and sufficient conditions for a link to be identifiable. (**Chapter 2**)

- For the second problem, we present a new network model that consists of bi-directional links and formally formulate constraints that capture the correlation among local links. We investigate how the correlation constraints help narrow down the bounds and develop a polynomial time algorithm, called Global ϵ-stabilizing (GES) algorithm, to calculate the best-effort bounds with link correlation constraints. We also find sufficient conditions where the best-effort bounds are the tightest. (**Chapter 3**)

- For the third problem, we prove that in the worst case we must list all possible measurement paths (MPs) in order to derive the global tightest error bounds[4]. In other words, there is no polynomial-time algorithm to derive the global tightest error bounds unless P=NP, since listing all measurement paths is $\#P$-complete [38]. Note that $\#P$-complete is at least as difficult as NP-complete [38]. In addition, we design a reinforcement learning (RL) based path construction method, called Guided Sequential Path Construction (*GSPC*). Quite different from traditional reinforcement learning methods, *GSPC* utilizes the special knowledge from our analysis and integrates both offline training over simulated networks and online prediction over the target network. This RL structure can effectively handle the difficulties of applying RL in the special application context of bandwidth tomography. (**Chapter 4**)

[4]A link error bound is called globally tightest if it is the smallest among all possible error bounds derived for the link with different sets of MPs.

- We perform an extensive evaluation of our solutions over real-world ISP topology as well as simulated networks. Compared with two baseline path construction methods, *Random* and Diversity Preferred (*DP*), *GSPC* improves *Random* and *DP* in terms of the average error bound by 238% and 193%, respectively. *GSPC* also returns near-optimal results in small-scale simulated networks where listing all measurement paths for deriving the ground-truth global optimum is possible. In addition, we extensively evaluate GES and compare its performance with that of the Calculate the Tightest Bounds(CTB) benchmark [25]. Evaluation results show that GES effectively leverages the extra constraint to achieve much tighter performance bounds than CTB. (**Chapter 5**)

The research of this thesis has led to the following publications:

- C.Y. Feng, J.W. An, K. Wu, and J.P. Wang, "Bound Inference and Reinforcement Learning-based Path Construction in Bandwidth Tomography," *IEEE Infocom*, May 2021.
- C.Y. Feng, J.W. An, K. Wu and J.P. Wang, "Bound Inference and Reinforcement Learning-Based Path Construction in Bandwidth Tomography," *IEEE/ACM Transactions on Networking*, vol. 30, no. 2, pp. 501-514, April 2022.
- J.W. An, C.Y. Feng, and K. Wu, "Improving Performance Bounds for Network min-Systems with Link Correlations," *IFIP Networking*, June 2024.

Chapter 2

The Basic Model and Solution

2.1 The Basic System Model

We model a network as a graph $\mathcal{G} = (V, L)$ that consists of $|V|$ nodes and $|L|$ links. With a set of monitors deployed at the nodes in the network, we can use existing methods, such as *pathload* [39], to measure the bandwidth of a measurement path (defined below). We are interested in the bandwidth of all links in the network. To simplify analysis, we assume that the maximum bandwidth over all links is b_{max}, which can be set based on the physical specification of the network. Note that the maximum bandwidth of a link cannot be higher than the link capacity. Also note that the analytical results of this dissertation are applicable for the scenario where different links have different maximum bandwidth values. The only change is that each link uses its own b_{max} instead of the "global" b_{max}.

Following the convention in network performance tomography [2,11], we introduce basic assumptions and notations as follows:

- $\mathcal{G}$: A connected and undirected graph. Each link has distinct end nodes (i.e., no self-loop), and no two links in $\mathcal{G}$ connect to the same pair of nodes.

- *Measurement path (MP)*: A non-loop path with only two monitors at its end nodes. For test purposes, probing packets along an MP could be routed via source routing. This assumption has been used in most existing work [2, 4]. While this assumption may be too strong in real-world networks, there are other alternative techniques based on software-defined networks (SDN) [40], which support a packet to travel through a specified path.

- The network under consideration is assumed to be “static", implying that either the bandwidth changes slowly relative to the measurement process or it represents statistical characteristics (e.g., mean) that stay constant within the time period under consideration. This assumption has been broadly adopted in most network tomography work [2, 4, 11, 14].

Relaxing the above assumptions would further complicate the already challenging problems in network tomography. To this end, we discuss potential directions for future research in Chapter 6 of this dissertation.

Fundamentally different from performance tomography with additive metrics [2, 41], bandwidth tomography uses *min*-operation, i.e., the bandwidth of an MP is the minimum bandwidth of all links along the MP. In other words, traditional linear algebra is not applicable for bandwidth tomography.

We use the example in Fig. 2.1 (a) to illustrate the concept. The network has three monitors marked in red, and there are three MPs among the monitors. Different MPs may lead to different results of end-to-end bandwidth. We use $x_{p,q}$, which is unknown, to denote the bandwidth on link $l_{p,q}(p, q = 1, 2, 3, 4, 5)$. If there is no link between node v_p and v_q, $x_{p,q} = 0$. b_i denotes the end-to-end bandwidth on MP $P_i(i = 1, 2, 3)$. Like most network tomography work [2, 4], we assume an undirected graph, i.e., $x_{p,q} = x_{q,p}$. Note that the results in this dissertation can be extended to directed networks by treating $x_{p,q}$ and $x_{q,p}$ to be different variables.

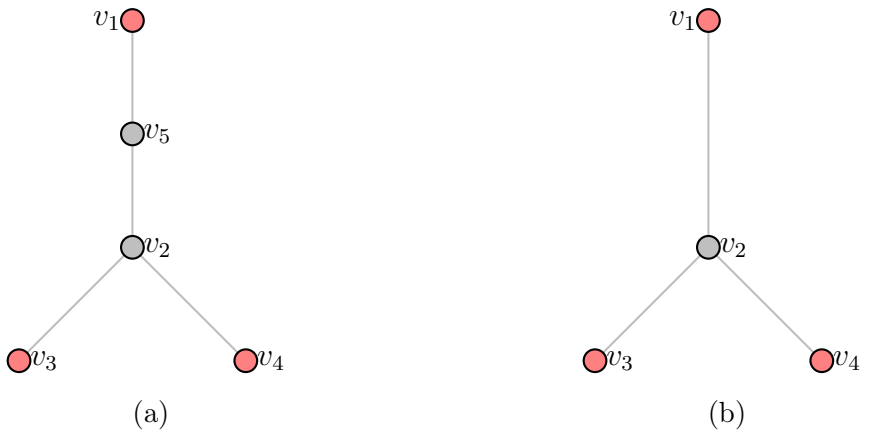


Figure 2.1: Example topology (a) and its simplified topology (b).

Due to the property of bandwidth, we have the following system of *min*-equations,

short-termed as *min*-system in the rest of the dissertation:

$$\begin{cases} x_{1,5} \wedge x_{5,2} \wedge x_{2,3} = b_1 \\ x_{1,5} \wedge x_{5,2} \wedge x_{2,4} = b_2 \\ x_{3,2} \wedge x_{2,4} = b_3 \end{cases} \tag{2.1}$$

where $\wedge$ means the *min* operation.

To simplify analysis, we can remove some non-monitor node of degree 2 (e.g., node v_5 in Fig. 2.1 (a)), and establish a virtual link between this node's two neighbors (e.g., nodes v_1 and v_2) if there is no link between them. This is because we have no way to find out the bandwidth value $x_{1,5}$ and the bandwidth value $x_{5,2}$ based on end-to-end measurements. In terms of bandwidth analysis, we can only infer the bandwidth on the path segment $v_1 \rightarrow v_5 \rightarrow v_2$. As such, we treat this path segment as a virtual link $l_{1,2}$. Note that if a link already exists between this node's two neighbors, we cannot remove this node and add the virtual link because otherwise the new graph will have two links connecting the same pair of nodes. The simplified network is shown in Fig. 2.1 (b). In addition, we should ignore any non-monitor node of degree 1 since there is no way to build an MP passing through this node. Therefore, in the rest of the dissertation, the target network $\mathcal{G}$ is the simplified network after the above pre-processing.

After the above pre-processing, the *min*-system for Fig. 2.1 (b) is

$$\begin{cases} x_{1,2} \wedge x_{2,3} = b_1 \\ x_{1,2} \wedge x_{2,4} = b_2 \\ x_{2,3} \wedge x_{2,4} = b_3 \end{cases} \tag{2.2}$$

With a slight abuse of notation $\wedge$, let's denote the above linear system into *equivalent* matrix form $\mathbf{R} \wedge \mathbf{x} = \mathbf{b}$, where

$$R = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \tag{2.3}$$

$$\mathbf{x} = \begin{pmatrix} x_{1,2} & x_{2,3} & x_{2,4} \end{pmatrix}^{\mathsf{T}} \tag{2.4}$$

$$\mathbf{b} = \begin{pmatrix} b_1 & b_2 & b_3 \end{pmatrix}^{\mathsf{T}} \tag{2.5}$$

The fundamental problem in bandwidth tomography is: **given R and b, can we infer the values of x?** We answer this question in the next section.

2.2 Bound Analysis and Algorithm for Obtaining the Tightest Error Bound

Different from the linear system with additive metrics [2, 4], the *min*-system cannot take advantage of existing results in linear algebra. Nevertheless, each *min*-equation can help to bound the bandwidth of related links. For instance, (2.2) can be formulated as:

$$\begin{cases} b_1 \le x_{1,2} \le b_{max}, b_1 \le x_{2,3} \le b_{max} \\ b_2 \le x_{1,2} \le b_{max}, b_2 \le x_{2,4} \le b_{max} \\ b_3 \le x_{2,3} \le b_{max}, b_3 \le x_{2,4} \le b_{max} \end{cases} \tag{2.6}$$

Hence, we can obtain the lower and upper bounds of the bandwidth for each link from the *min*-system. Define the *error bound* as the gap between the upper bound and the lower bound. Our goal is to reduce the above naïve error bound as much as possible. That is, if a link is identifiable, the error bound of its bandwidth should be 0; otherwise, its error bound should be the tightest. Note that the tightest error bound in this section is *conditional* in the sense that it is the best bound that we can derive from the given *min*-system.

To solve the problem, we have the following observation: a *min*-equation of k variables $\wedge_{i=1}^{k} x_i = b$ is equivalent to the following two conditions:

$$\begin{cases} (1)\ b \le x_i \le b_{max}, i = 1, \ldots, k \\ (2) \text{ at least one of } x_i \text{ is equal to } b. \end{cases} \tag{2.7}$$

Based on this observation, we can prove the following lemma:

Lemma 1. *Assume that a min-system has n variables and m min-equations, $\mathbf{R} \wedge \mathbf{x} = \mathbf{b}$, where $\mathbf{R}$ is an $m \times n$ Boolean matrix, $\mathbf{x} = \begin{pmatrix} x_1 & \ldots & x_n \end{pmatrix}^\intercal$, and $\mathbf{b} = \begin{pmatrix} b_1 & \ldots & b_m \end{pmatrix}^\intercal$. Assume that the number of distinct $b_i (i = 1, \ldots, m)$ is d. W.L.O.G., assume that $b_i \le b_j (i \le j)$ and the d distinct values are $b'_1, \ldots, b'_d$, respectively. We have:*

1. *The links corresponding to $\mathbf{x}$ can be divided into d disjoint nonempty sets, de-*

noted by $S_{b'_k}(k = 1, \ldots, d)$.

2. *Furthermore, if* $|S_{b'_k}| = 1$*, then the link in* $S_{b'_k}$ *is identifiable (i.e., its error bound is zero), otherwise,* b'_k *is the greatest lower bound for the links in* $S_{b'_k}$.

Proof. Base on the first condition in (2.7), we can build d intervals $[b'_1, b_{max}]$, $[b'_2, b_{max}]$, ..., $[b'_d, b_{max}]$, with each interval corresponding to a set denoted by $S_{b'_1}, S_{b'_2}, \ldots, S_{b'_d}$, respectively. Note that the subscribes $b'_1, \ldots, b'_d$ of the sets denote the bandwidth value for clearly reflecting the correspondence between the intervals and the sets. Also note that each set records links, whose bandwidth values fall in the set's corresponding interval after the *unique* assignment described below.

To prove (1), we only need to show that based on the *min*-system $\mathbf{R} \wedge \mathbf{x} = \mathbf{b}$ we can find a way to *uniquely* assign any $x_i \in \mathbf{x}$ to one of the d intervals. The method to assign x_i into an interval is as follows: If x_i appears in l *min*-equations, whose values are ordered in non-decreasing order and are $b'_{j_1}, \ldots, b'_{j_l}$, respectively, then we put x_i into the interval $[b'_{j_l}, b_{max}]$. Clearly, such an assignment is unique for x_i. Based on the second condition of (2.7), each interval must include at least one measurement value and hence (1) is proved. Since any measurement value in $[b'_i, b_{max}]$ must also fall in $[b'_j, b_{max}]$ where $b'_j < b'_i$, the above assignment implies the greatest lower bound of x_i that we can obtain from the *min*-system.

Based on the second condition in (2.7), every interval must include at least one measurement value (i.e., correspondingly one link). If an interval $[b'_k, b_{max}]$ only includes one measurement value, the link corresponding to this value is hence identified. If the interval includes multiple measurement values, however, we cannot identify all the corresponding links. In this case, b'_k is the greatest lower bound for these links. Hence, (2) is proved. □

The proof of Lemma 1 is constructive (w.r.t the link-interval assignment), based on which we can design an Algorithm, called CTB, to find the tightest error bound for every link in the given *min*-system. The pseudo-code is shown in Algorithm 1. Algorithm 1 is correct due to Lemma 1 and the fact that if a link is not identifiable, then the *min*-system offers no information to reduce its upper bound. The worst-case complexity of Algorithm 1 is $O(mn + mlogm)$, since the first step takes $O(m \log m)$ and the rest of code is mainly two loops over m and n.

Lemma 1 and CTB are important to understand the rest analysis in the dissertation. CTB works in a way similar to Gaussian elimination. We give an example to

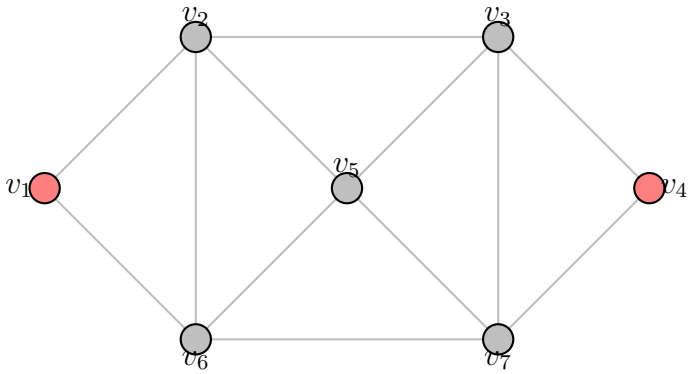


Figure 2.2: An example network to illustrate CTB.

illustrate the operations in CTB so that readers can understand the intuition of the lemmas and theorems in the rest of the dissertation.

Example 1. ***(Example of CTB)*** *An example network topology is shown in Fig. 2.2, where two monitors are marked in red. Assume that we have built four MPs and their corresponding min-system is shown below:*

$$\begin{cases} P_1 : x_{1,2} \wedge x_{2,3} \wedge x_{3,4} = 2 \\ P_2 : x_{1,6} \wedge x_{6,7} \wedge x_{7,4} = 1 \\ P_3 : x_{1,6} \wedge x_{6,5} \wedge x_{5,7} \wedge x_{7,4} = 2 \\ P_4 : x_{1,2} \wedge x_{2,6} \wedge x_{6,7} \wedge x_{7,4} = 1 \end{cases} \tag{2.8}$$

This min-system only has two distinct end-to-end bandwidth values and the maximum is 2*. This means that the greatest lower bound for the covered links is at most* 2*. Therefore, in order to find out the links with the greatest lower bound* 2*, we sort the min-equations in the non-increasing order based on their values.*

$$R = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \tag{2.9}$$

$$\mathbf{x} = \begin{pmatrix} x_{1,2} & x_{2,3} & x_{3,4} & x_{1,6} & x_{6,5} & x_{5,7} & x_{7,4} & x_{6,7} & x_{2,6} \end{pmatrix}^{\intercal} \tag{2.10}$$

$$\mathbf{b} = \begin{pmatrix} 2 & 2 & 1 & 1 \end{pmatrix}^{\mathsf{T}} \tag{2.11}$$

Now, we consider the paths with the maximum end-to-end bandwidth (i.e., P_1 and P_3). Obviously, the links on these paths cannot be identified. Nevertheless, we can conclude that all links covered by the two paths must have a bandwidth no smaller than 2. We thus can set the lower bandwidth bound of these links to 2, and in the meantime, we do not need to consider these links anymore (because they have been assigned to the corresponding set S_2, where the subscript 2 denotes the bandwidth value). After removing these links, we have the following new min-system.

$$R_{new} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \tag{2.12}$$

$$\mathbf{x}_{new} = \begin{pmatrix} x_{6,7} & x_{2,6} \end{pmatrix}^{\mathsf{T}} \tag{2.13}$$

$$\mathbf{b}_{new} = \begin{pmatrix} 1 & 1 \end{pmatrix}^{\mathsf{T}} \tag{2.14}$$

The above process is equivalent to the variable elimination part in CTB. In this new min-system, all the "paths" have the same end-to-end bandwidth 1. Because all the links with higher lower bound on P_2 (i.e., $l_{1,6}$ and $l_{7,4}$) have been removed, it's clear that $l_{6,7}$ is identifiable and its value $x_{6,7}$ is 1. This is the only link that can be identified by the original min-system.

The rest part of CTB (Line 11-Line 22) is to find out all the identifiable links in the min-system and divide other unidentifiable links into the disjoint sets according to their greatest lower bound. In this example, all the covered links can be divided into two disjoint nonempty sets S_1 and S_2. The final result is shown in Table 2.1.

Table 2.1: Dividing links into disjoint sets

	Covered links
S_2	$l_{1,2}, l_{2,3}, l_{3,4}, l_{1,6}, l_{6,5}, l_{5,7}, l_{7,4}$
S_1	$l^*_{6,7}, l_{2,6}$

Note: * means identifiable.

Based on CTB, we can prove the necessary and sufficient condition for a link to be identifiable. The sufficient condition is straightforward from Lemma 1. The necessary condition can be easily proved with contradiction.

Theorem 1. *Assume that a min-system has n variables and m min-equations, $\mathbf{R} \wedge \mathbf{x} = \mathbf{b}$, where $\mathbf{R}$ is an $m \times n$ Boolean matrix, $\mathbf{x} = \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix}^\intercal$, and $\mathbf{b} = \begin{pmatrix} b_1 & \dots & b_m \end{pmatrix}^\intercal$. A link l is identifiable in this min-system iff there is a path P containing link l with end-to-end bandwidth b_P, such that $\forall l' \in P \setminus l$, the greatest lower bandwidth bound of l' derived from CTB is $b_{l'}$ where $b_{l'} > b_P$.*

Proof. ($\Leftarrow$) As we mentioned above, a path P whose end-to-end bandwidth is b_P not only provides the lower bound of its covered links, but also has at least one link whose bandwidth is b_P. So, if all the links in P except l obtain a larger lower bandwidth bound derived from CTB, the exact bandwidth of link l must be b_P, i.e., l can be identified.

($\Rightarrow$) If link l is identifiable by a *min*-system, then its error bound derived from the *min*-system is 0. During the variable elimination process, CTB gradually removes links with their highest lower bounds in each *min*-equation. If all the processed *min*-equations containing link l have at least two links after the variable elimination, we cannot determine that the error bound of link l is 0. Thus, there must exist at least one processed *min*-equation that only contains link l after variable elimination. Denote this *min*-equation's corresponding path as P. The fact that this processed *min*-equation only contains l is equivalent to the statement that $\forall l' \in P \setminus l$, the highest lower bandwidth bound of l' derived from CTB is $b_{l'}$ where $b_{l'} > b_P$. □

Algorithm 1: Calculate the Tightest Bounds (CTB)

input : a *min*-system, $\mathbf{R} \wedge \mathbf{x} = \mathbf{b}$, where $\mathbf{R} = [r_{ij}]_{m \times n}$ is a Boolean matrix, $\mathbf{x} = \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix}^\intercal$, and $\mathbf{b} = \begin{pmatrix} b_1 & \dots & b_m \end{pmatrix}^\intercal$

output: the tightest error bound for every link

```
1  begin
2      Order the m min-equations in the non-increasing order of their values;
       /* Variable elimination (like Gaussian elimination)        */
3      for i = 1 to m − 1 do
4          for j = i + 1 to m do
5              if b_i == b_j then
6                  continue;
7              end
8              r_jk = max{r_jk − r_ik, 0}, k = 1, ..., n;
9          end
10     end
       /* Assigning interval and determine bounds                 */
11     for i = 1 to m do
12         for j = 1 to n do
13             if r_ij == 1 then
14                 Assign x_j into interval [b_i, b_max], i.e., x_j's error bound is
                   b_max − b_i;
15             end
16         end
17     end
18     for i = 1 to m do
19         if there is only one r_ij == 1(j = 1, ..., n) then
20             x_j = b_i, i.e., x_j's error bound reduces to 0;
21         end
22     end
23 end
```

Chapter 3

The Constrained Model and Solution

3.1 Motivation

Following the convention, when the value of a link can be uniquely determined, we call the link *identifiable* [37]. When a link is not identifiable, we aim to discover the tightest bound of each link. From Chapter 2, we know that solving a network *min*-system is extremely challenging because the *min* operator results in high information loss: it only tells the minimum value among the links along a path, based on which it is impossible to recover all the values on this path's constituting links. Thus, a reasonable goal is to find the tightest bound of each link determined by the *min*-system.

In Chapter 2, we have developed a polynomial algorithm to obtain the tightest bounds on variables in a *min*-system. Nevertheless, the theoretically proven tightest bounds may still be too loose. Specifically, a link's upper bound is its maximum capacity. In the context of bandwidth tomography, this is equivalent to stating that the available bandwidth of a link is smaller than the link's maximum capacity, which is true but nevertheless not useful. An interesting question arises: **can we introduce extra information into the *min*-system to refine these bounds?**

The extra information that we consider includes two constraints: the fairness constraint and the total capacity constraint for links adjacent to a common node. The former means that the performance of these links should not be drastically different; the latter means that the total values on these links are bounded. These constraints normally hold in network systems, with the former implying that no link should be severely over-utilized compared to other links adjacent to the common node and

the latter implying that the processing capacity of a node is limited. It is worth noting that the above constraints resemble many real-world scenarios. In the Internet backbone, most routers adopt load balancing that dispatches a traffic flow among different links of the router. In this case, the bandwidth of links behind the load balancing is highly correlated. In some hybrid wired-wireless networks in the era of the Internet of Things (IoT) and edge computing [42–46], the performance of wireless links of a gateway node is correlated since the links share the same Radio Frequency (RF) channel.

Solving network *min*-system with correlated links poses several non-trivial difficulties: First, one basic assumption in existing work that links are independent does not hold anymore. Second, assuming bi-directional links[1] may not be appropriate when we consider the above constraints. For instance, in the wireless scenario, the uplinks and downlinks may use different RF channels and thus form different interference link sets. Third, an update on one link's performance may generate a cascading effect on other links' performance, and we might obtain different final results when updating link performance in different orders.

This chapter aims to address all the above challenges.

3.2 The Constrained System Model

We present a generic abstract network model that captures the link correlation that exists in many scenarios. The network is modelled as a directed graph $\mathcal{G} = (V, L)$, where V denotes the set of nodes (e.g., computers or routers), L denotes the set of links between nodes. Note that for two nodes v_1 and v_2, the performance value from v_1 to v_2, denoted by l_{12}, may be different from that from v_2 to v_1, denoted by l_{21}. We call the link between v_1 and v_2 bi-directional when $l_{12} = l_{21}$, otherwise, we call it uni-directional. In summary, $\mathcal{G}$ is a connected and directed graph. Each link has distinct end nodes (i.e., no self-loop), and no two links of the same direction in $\mathcal{G}$ connect to the same pair of nodes. We explain our system model using bandwidth tomography as the application context to ease discussion.

With a set of monitors deployed at some nodes in the network, we assume that we can measure the end-to-end performance of a *Measurement path (MP)*. Here, an MP refers to a non-loop path that only contains two monitors at its end nodes. We

[1] A bi-directional link between node A and node B means that the performance from node A to node B is equal to that from B to A. Otherwise, we call the link uni-directional.

can use existing methods, such as *pathload* [39] and RT-WABest [47], to measure the end-to-end available bandwidth of an MP. Here, we are interested in inferring the available bandwidth (or **bandwidth for short**) of all links in the network based on measurement results along MPs. To simplify the analysis, we assume that the maximum bandwidth over all links is b_{max}. Note that the analytical results of this dissertation are easily applicable to the scenario where different links have different maximum bandwidth values. The only change is that each link uses its own maximum value. Initially, b_{max} could be set to the physical limit of the link based on the hardware specification.

The bandwidth values in a local environment may be correlated. As shown in Fig. 3.1, if routers v_5 and v_6 are equipped with a load balancer to balance the traffic load towards the connected servers, then l_{51} and l_{52} are correlated, and l_{63} and l_{64} are also correlated. Suppose that our network under analysis is a hybrid wireless network that consists of both wired and wireless links. In a local area (e.g., base station), the wireless uplinks may share the same radio channel and thus their bandwidth values are correlated; similarly, the wireless downlinks may share another radio channel and thus form another set of correlated links [48]. We next propose constraints that can be used to analyze all the above scenarios where links sharing common nodes are correlated. For this purpose, we consider two constraints that have broad practical implications. To simplify notation, we use the same notation to denote a link and the bandwidth value of the link.

Definition 1. ***Total capacity constraint:*** *For a subset of links connected to a node v, called correlation set and denoted by $\{l_1^v, l_2^v, \ldots, l_m^v\}$, we assume that:*

$$\sum_{i=1}^{m} l_i^v \leq b_{max}^v. \tag{3.1}$$

This constraint is based on the observation that a router's total packet processing speed is limited. This constraint is also applicable to the wireless case, where the wireless links share the same radio channel, and their total bandwidth should be no larger than the capacity of the channel.

Definition 2. ***Fairness constraint:*** *For a subset of links connected to a node v,*

called correlation set and denoted by $\{l_1^v, l_2^v, \ldots, l_m^v\}$*, we assume that:*

$$\mathcal{J}(l_1^v, \ldots, l_m^v) \equiv \frac{(\sum_{i=1}^m l_i^v)^2}{m \sum_{i=1}^m (l_i^v)^2} \geq \delta_v. \tag{3.2}$$

Eq. (3.2) means Jain's fairness index [49] is higher than a threshold, reflecting the fact that load balancing in a router tries to avoid overloading a particular outgoing link. This constraint is also reasonable to wireless networks since most wireless systems have mechanisms to guarantee a certain level of fair share of the channel among the local wireless links [50,51]. Note that Jain's fairness index is a value between 0 and 1, and a higher value means more fair. If the index is 1, all links (in the correlation set) have the same value. As the link disparity increases, the index decreases.

Remark 1. *The constraints are optional, i.e., they are posed only to the nodes that we know have these constraints. In addition, for a given node* v*, the correlation set for the total capacity constraint and the correlation set for the fairness constraint are not necessarily the same.*

Remark 2. *We assume that the network under consideration is "static", implying that either the bandwidth changes slowly relative to the measurement process or it represents statistical characteristics (e.g., mean) that stay constant within the time period under consideration. This assumption has been broadly adopted in most network tomography work [2, 4, 11, 14]. It has been observed that the available bandwidth of wireless links can be considered stationary [52] over a short time period, even if, in general, it is dynamic Quality of Service(QoS) information.*

In the following, we use a simple network in Fig. 3.1 to illustrate the network *min*-system with correlation constraints. Assume that three monitors are deployed at nodes v_1, v_2 and v_3, respectively. There are six possible MPs in total, which are (1) $v_1 \to v_5 \to v_2$, (2) $v_2 \to v_5 \to v_1$, (3) $v_1 \to v_5 \to v_6 \to v_3$, (4) $v_3 \to v_6 \to v_5 \to v_1$, (5) $v_2 \to v_5 \to v_6 \to v_3$, and (6) $v_3 \to v_6 \to v_5 \to v_2$. Assume that the end-to-end available bandwidth of the above 6 MPs is $b_1, b_2, \ldots, b_6$, respectively.

Since the bandwidth of a path is the minimum bandwidth of all links along the

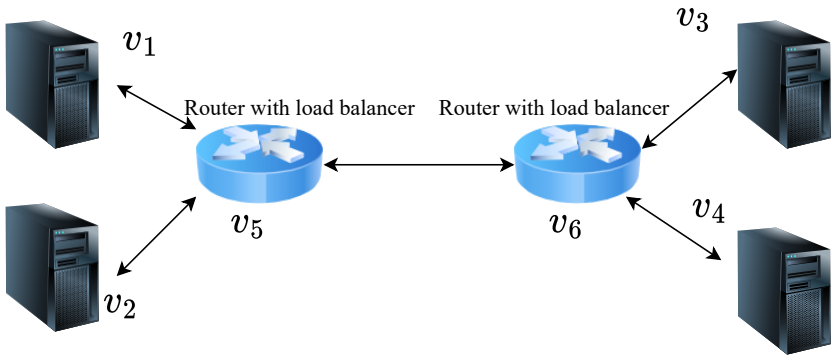


Figure 3.1: An example network with correlated links caused by load balancing. v_1, v_2, and v_3 are also monitoring nodes.

path, we have the following system of *min*-equations:

$$\begin{cases} l_{15} \wedge l_{52} = b_1 \\ l_{25} \wedge l_{51} = b_2 \\ l_{15} \wedge l_{56} \wedge l_{63} = b_3 \\ l_{36} \wedge l_{65} \wedge l_{51} = b_4 \\ l_{25} \wedge l_{56} \wedge l_{63} = b_5 \\ l_{36} \wedge l_{65} \wedge l_{52} = b_6 \end{cases} \tag{3.3}$$

where l_{ij} denotes the bandwidth from node v_i to node v_j and $\wedge$ means the *min* operation.

With a slight abuse of notation $\wedge$, let's denote the above linear system into *equivalent* matrix form $\mathbf{R} \wedge \mathbf{L} = \mathbf{B}$, where

$$\mathbf{R} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \tag{3.4}$$

$$\mathbf{L} = \left(l_{15}, l_{51}, l_{25}, l_{52}, l_{36}, l_{63}, l_{46}, l_{64}, l_{56}, l_{65}\right)^{\mathsf{T}}, \tag{3.5}$$

$$\mathbf{B} = \begin{pmatrix} b_1 & b_2 & b_3 & b_4 & b_5 & b_6 \end{pmatrix}^{\mathsf{T}}. \tag{3.6}$$

If we know v_5 poses load balancing on links l_{52} and l_{51} and v_6 poses load balancing on links l_{63} and l_{64}. We then have the following fairness constraints due to the load balancing in v_5 and v_6:

$$\begin{cases} \mathcal{J}(l_{51}, l_{52}) \geq \delta_{v_5} \\ \mathcal{J}(l_{63}, l_{64}) \geq \delta_{v_6} \end{cases}$$

If we know that v_5 has a control mechanism that limits the total bandwidth to/from v_1 and v_2, and v_6 has a control mechanism that limits the total bandwidth to/from v_3 and v_4, we have the following total capacity constraints:

$$\begin{cases} l_{51} + l_{52} \leq b_{max}^{v_5\downarrow} \\ l_{63} + l_{64} \leq b_{max}^{v_6\downarrow} \\ l_{15} + l_{25} \leq b_{max}^{v_5\uparrow} \\ l_{36} + l_{46} \leq b_{max}^{v_6\uparrow} \end{cases}$$

where different links' total capacities differentiate $b_{max}^{v_i\downarrow}$ from $b_{max}^{v_i\uparrow}$. For a link l, if we can determine its exact value, we call the link identifiable. Otherwise, we can only determine an interval, also called error bound, that covers its value. This dissertation aims to answer the following fundamental problem: **given R, B and the aforementioned constraints, can we infer the exact values or the tightest error bounds of L?**

3.3 Bound Analysis and Algorithm for Obtaining the Tightest Error Bound

3.3.1 Elaboration on the Tightest Error Bound

In Chapter 2, an algorithm called CTB was proposed to calculate the "tightest" error bounds of **L**. Nevertheless, since the CTB algorithm assumes no link correlation information is available and does not change the upper bound of link values, CTB returns the highest lower bound based only on the set of *min*-equations. Nevertheless, the new link correlation information offers us new opportunities to further reduce the

Table 3.1: Main Notations Used in This Chapter

Notation	**Explanation**
$\mathcal{L}_{all}$	The set of all the links in the *min*-system
$\mathcal{N}_{all}$	The set of all the nodes that are passed by at least one measurement path
$\mathcal{L}_v$	The set of all links involved in the same constraints of specific (3.7) or (3.8) at node v
$\mathcal{L}_v^I$	The set of all the current identifiable links in $\mathcal{L}_v$ at node v
$\mathcal{L}_v \setminus \mathcal{L}_v^I$	The set of all the current unidentifiable links in $\mathcal{L}_v$ at node v
α_v	A parameter defined in (3.11) for convexity check at node v

error bound, i.e., raising the lower bound and decreasing the upper bound of an unidentifiable link. In addition, some links may become identifiable due to these additional constraints. Accordingly, we should first make clear the exact meaning of the tightest error bound in our new context.

Definition 3. *(**Tightest error bound**): In the network min-system with correlated links, the tightest error bound of an unidentifiable link is defined as the smallest interval containing all possible values the link may have, satisfying the min-system and all the correlation constraints.*

3.3.2 Challenges in Obtaining the Tightest Error Bounds

Obviously, considering all links' correlation constraints at the same time complicates the solution. It's worth noting that each correlation constraint only applies to the links associating a local node. Thus, firstly, we should investigate how the bound information of an unidentifiable link l_1^v could be influenced by the correlation constraints associated with the node v.

Clearly, to obtain the tightest error bound for l_1^v, we need to seek its worst-case minimum value and the worst-case maximum value using the directly-related correlation constraints. In other words, if putting the *min*-equations aside, we need to solve the following two optimization problems (3.7) and (3.8), respectively.

$$\begin{aligned} \min \quad & l_1^v \\ \text{s.t.} \quad & \frac{(\sum_{i=1}^m l_i^v)^2}{m\sum_{i=1}^m l_i^{v2}} \geq \delta_v, \\ & \sum_{i=1}^m l_i^v \leq b_{max}^v, \\ & l_i^v \in [\hat{l}_{i-}^v, \hat{l}_{i+}^v], i = 1, \cdots, m, \end{aligned} \tag{3.7}$$

and

$$\begin{aligned} \min \quad & -l_1^v \\ \text{s.t.} \quad & \textit{the same constraints as } (3.7), \end{aligned} \tag{3.8}$$

where $l_i^v (i = 1, \cdots, m)$ are all links having constraints associated with node v, the fairness parameter is δ_v, the total capacity parameter is b_{max}^v, and $\hat{l}_{i-}^v$ and $\hat{l}_{i+}^v$ are link l_i^v's *current* lower bound and upper bound, respectively. Then, we take the *min*-

system into consideration. Specifically, the *current* lower and upper bounds can be obtained with the CTB algorithm [25].

Essentially, the solutions to Problems (3.7) and (3.8) will be the basic building blocks to infer the tightest error bounds for all unidentifiable links in the min-system with constraints. Even having the above building blocks, however, we still need to answer two challenging problems:

1. **Challenge 1:** Synchronous vs asynchronous local updates. For a given link, we can update its lower and upper bounds synchronously (i.e., solve Problems (3.7) and (3.8) in parallel) or asynchronously (i.e., solving Problems (3.7) and (3.8) in sequential. When a node is connected with multiple links, we can update the upper and lower bounds of these links synchronously (i.e., update all links' bounds in parallel) or asynchronously (update them sequentially). Will different ways of updating bounds lead to different answers for links associated with the same node?

2. **Challenge 2:** Cascading impact: In the whole network, when a node locally updates the bounds of its associated links, the results may impact the link-bound updates of other nodes. Will the update order among different nodes lead to different final answers?

It is worth noting that Challenge 1 concerns the bound update order *locally*, i.e., updating links associated with a local node, and Challenge 2 concerns the bound update order *globally*, i.e., the order for the node-by-node update. In the following, we theoretically analyze the condition that leads to unique, tightest error bounds for local updates resilient to synchronous/asynchronous updates. We design an algorithm that leads to the guaranteed tightest error bounds when the condition holds and can empirically improve the error bounds of CTB when the condition does not hold.

3.4 Tackling Challenge 1: The CBLN algorithm and the Local Tightest Error Bounds

Since the total capacity constraint is linear, the main difficulty of solving Problems (3.7) and (3.8) comes from the fairness constraint. In the first part of this section, we analyze the general properties of the fairness constraint in (3.7) and (3.8).

We can transform the fairness constraint into the following form:

$$f(l_1^v, ..., l_m^v) = \delta_v m(l_1^{v\,2} + ... + l_m^{v\,2}) - (l_1^v + ... + l_m^v)^2 = \mathbf{l}^T H \mathbf{l} \leq 0, \tag{3.9}$$

where $\mathbf{l} = (l_1^v, ..., l_m^v)^T$, $m \geq 2$, and

$$H = \begin{pmatrix} \delta_v m - 1 & -1 & -1 & \dots & -1 \\ -1 & \delta_v m - 1 & -1 & \dots & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ -1 & -1 & -1 & \dots & \delta_v m - 1 \end{pmatrix}.$$

H is an $\mathbf{R}^{m \times m}$ symmetric matrix. The elements of H are all -1 except that the diagonal elements are all $\delta_v m - 1$. We ignore the case $m = 1$ because it is meaningless to discuss fairness if there is only one link.

Note that during the process of bounds update over the whole *min*-system, an unidentifiable link may become identifiable. To ease discussion with a unified presentation, we extend the constraint (3.9) into a more general form that reflects the fairness constraint changes during the bounds update process. To be specific, suppose $\mathcal{L}_v$ consists of all the links involved in the same constraints of (3.7) or (3.8) at node v and $|\mathcal{L}_v| = m$, when $n (1 \leq n \leq m-1)$ links become identifiable at node v, we record these links in set $\mathcal{L}_v^I$ and denote their corresponding values as $I_k^v (k = 1, \cdots, n)$. We also record the rest unidentifiable links in set $\mathcal{L}_v \setminus \mathcal{L}_v^I$. The fairness constraint (3.9) for the links in $\mathcal{L}_v \setminus \mathcal{L}_v^I$ could be revised to

$$f(l_1^v, ..., l_{m-n}^v) = \mathbf{l}^T H' \mathbf{l} - 2S_1 \mathbf{l}^T \mathbf{1} + \delta_v m S_2 - S_1^2 \leq 0, \tag{3.10}$$

where $\mathbf{l}^T = (l_1^v, ..., l_{m-n}^v)$, $\mathbf{1} = (1, ..., 1)^T$ has the same length as $\mathbf{l}$, $S_1 = \sum_{l_i^v \in \mathcal{L}_v^I} I_i^v$, $S_2 = \sum_{l_j^v \in \mathcal{L}_v^I} I_j^{v\,2}$, and H', which is similar to H except the dimension, is an $\mathbf{R}^{(m-n)\times(m-n)}$ symmetric matrix whose elements are all -1 except the diagonal all $\delta_v m - 1$.

Since H' is a special pattern of the circulant matrix, we could construct a series of eigenvectors $\mathbf{v}_i (i = 1, ..., m-n) \in \mathbf{R}^{(m-n)\times 1}$ where $\mathbf{v}_1 = \mathbf{1}$ and $\mathbf{v}_j = \mathbf{e}_j - \mathbf{e}_1 (j = 2, ..., m-n$ and $\mathbf{e}_i$ is the ith column of the identity matrix $\mathbf{I}_{m-n})$. Then, it's easy to show that $|H' - \lambda I| = (\delta_v m - \lambda)^{m-n-1}(\delta_v m - m + n - \lambda)$, and the eigenvalues of H' are $\delta_v m$ and $\delta_v m - m + n$. Because $\delta_v m > 0$, we denote:

$$\alpha_v = \delta_v m - m + n. \tag{3.11}$$

When $\alpha_v \geq 0$, H' is a positive semi-definite matrix, and Problems (3.7) and (3.8) for all constrained links are all **convex**. In this case, existing convex programming tools (i.e. GUROBI, MOSEK) could help search the globally unique optimal solutions under given accuracy in pseudo-polynomial time. Later, we prove that we could obtain the *local* tightest error bound in this case.

However, when $\alpha_v < 0$, the fairness constraint constraint of (3.7) and (3.8) would not be convex anymore. If we want to obtain the *local* tightest error bound at v, i.e., the smallest interval containing all possible values the link l_i may have and satisfying the min-system and all the *related* correlation constraints, the globally optimal solutions of non-convex version of (3.7) and (3.8) must be required. In this case, when we only need to consider the local updating influence of links connecting to a node v, we may check every local optimal solution of optimizations. But this would be extremely time-consuming or even computationally untractable due to the nonconvexity and the need to consider all the nodes of *min*-system.

Note that in $\alpha_v = \delta_v m - m + n$, n is the number of identifiable links. When we consider one link's updating influence on the whole *min*-system (i.e., global updating influence), one link's local updating from (3.7) and (3.8) at node v might change other links, which are not constrained by v, from unidentifiable to identifiable. Therefore, when δ_v and m are fixed, α_v of all nodes is increasing during the whole *min*-system updating process. For an efficient global nodes' updating process, we would ignore the fairness constraint and only consider the total capacity constraint for local updates in the case $\alpha_v < 0$.

To ease the later discussion, we relabel the version of Problems (3.7) and (3.8), after removing the fairness constraints, as Problems (3.7.a) and (3.8.a). Then, we have the following lemma.

Lemma 2. *Assume that the bounds information of all links in $\mathcal{L}_{all} \setminus \mathcal{L}_v$ are fixed. For $\forall l_i \in \mathcal{L}_v \setminus \mathcal{L}_v^I$, the optimal solutions of Problem (3.7.a) would not update its lower bound.*

Proof. The above lemma is straightforward: In the constraints of (3.7.a) and (3.8.a), we only have the total capacity constraint and the current bounds interval $[\hat{l}_{i-}, \hat{l}_{i+}]$ for each involved unidentifiable link $l_i \in \mathcal{L}_v \setminus \mathcal{L}_v^I$. Because each link's current bounds interval must contain its real bandwidth value, the total capacity constraint could always be satisfied when $l_i = \hat{l}_{i-}$. □

Further, we analyze how (3.8.a) would change each l_i's upper bound. Obviously,

the upper bound of l_i provided by the total capacity constraint is obtained when all the other unidentifiable links are equal to their current lower bounds. Thus, the computing and updating processes of (3.7.a) and (3.8.a) are independent. Based on the above analysis, we design Algorithm 2 (CBLN(l_i, v)), which calculates the new bound for an unidentifiable link l_i at node v. Essentially, when $\alpha_v \geq 0$, CBLN(l_i, v) updates the bounds using both the fairness and total capacity constraints; otherwise, it updates the bounds only using the total capacity constraint. Note that in the CBLN(l_i, v) algorithm, we not only return the new bounds $[\hat{l}'_{i-}, \hat{l}'_{i+}]$ for link l_i, but also record the values of all other links when l_i obtains its new lower bound $\hat{l}'_{i-}$ by Problem (3.7) or (3.7.a) and the values of all other links when l_i obtains its new upper bound $\hat{l}'_{i+}$ by Problem (3.8) or (3.8.a). We use $\mathcal{C}_{l_{i-}}(v)$ and $\mathcal{C}_{l_{i+}}(v)$ to denote them, respectively. In Section 3.6, $\mathcal{C}_{l_{i-}}(v)$ and $\mathcal{C}_{l_{i+}}(v)$ will help analyze the updating impact, caused by running CBLN algorithms derived from other nodes, to the optimal bounds of l_i obtained at v.

In the second part of this section, we demonstrate a good property of the Algorithm 2 (CBLN(l_i, v)). Due to the design of the optimization problems (3.7) and (3.8) and the features of the fairness and total capacity constraints, we have the following lemma:

Lemma 3. *Assume that the bounds information of all links in $\mathcal{L}_{all} \setminus \mathcal{L}_v$ are fixed. For $\forall l_i, l_j \in \mathcal{L}_v \setminus \mathcal{L}_v^I$, the output of CBLN($l_i$, v) would not impact the output of CBLN(l_j, v).*

Proof. The assumption that all the bounds information of all links in $\mathcal{L}_{all} \setminus \mathcal{L}_v$ are fixed allows us to focus on investigating the local effect caused by running CBLN(l_i, v) to the other unidentifiable links at local node v, i.e. links in $\mathcal{L}_v \setminus \mathcal{L}_v^I$.

Without loss of generality, suppose that we firstly run CBLN(l_i, v) and secondly run CBLN(l_j, v) and finally run CBLN(l_i, v) again and keep the links' bounds information, i.e. the corresponding outputs of each CBLN run, in $\mathcal{L}_{all} \setminus \mathcal{L}_v$ updating during the whole process. Then, we can prove the following two facts:

- The local bound information updates caused by the first run of CBLN(l_i, v) would not influence the output of CBLN(l_j, v), i.e. $\mathcal{C}_{l_{j-}}(v)$, $\mathcal{C}_{l_{j+}}(v)$ and the newly obtained bandwidth bound $[\hat{l}'_{j-}, \hat{l}'_{j+}]$;
- After running CBLN(l_j, v), the output of the second run of CBLN(l_i, v) would be the same as the first run of CBLN(l_i, v).

Proof of the first fact: In the case where $\alpha_v < 0$, according to lemma 2. $\forall l_i \in \mathcal{L}_v \setminus \mathcal{L}_v^I$, CBLN($l_i$, v) would not change its lower bound. Thus, the updating order would make no difference to the lower bound of every unidentifiable link l_k and its $\mathcal{C}_{l_{k-}}(v)$, which would keep its original setting. In addition, the upper bound of l_k provided by any CBLN(l_k, v) would only be determined by all other links' lower bounds, and $\mathcal{C}_{l_{k+}}(v)$ also records all other links' lower bounds. Because the updating order of links would not influence their lower bounds, the output of (3.8.a) for every l_k would not be influenced either.

Then, let's turn to the case where $\alpha_v \geq 0$ and prove the first fact. Suppose that the output of the first run of CBLN(l_i, v) is $\mathcal{C}_{l_{i-}}^{(1)}(v)$, $\mathcal{C}_{l_{i+}}^{(1)}(v)$ and $[\hat{l}_{i-}^{(1)}, \hat{l}_{i+}^{(1)}]$; the output of the CBLN(l_j, v) is $\mathcal{C}_{l_{j-}}(v)$, $\mathcal{C}_{l_{j+}}(v)$ and $[\hat{l}_{j-}, \hat{l}_{j+}]$. When $\alpha_v \geq 0$, $[\hat{l}_{i-}^{(1)}, \hat{l}_{i+}^{(1)}]$ is the feasible interval of l_i determined by the convex set which is generated by the constraints of (3.7). That is to say, all the possible values l_i that can take in that convex set form an interval is exactly $[\hat{l}_{i-}^{(1)}, \hat{l}_{i+}^{(1)}]$. Although CBLN($l_i$, v) was calculated before and updated the current bound of l_i in the constraints of CBLN(l_j, v). They actually refer to the same convex set. Thus, the first run of CBLN(l_i, v) would not influence the optimal values l_j taken in CBLN(l_j, v). If multiple values for l_i could make l_j to reach its optimal solutions of related (3.7) and (3.8) in the process CBLN(l_j, v), they must be all included in its feasible interval $[\hat{l}_{i-}^{(1)}, \hat{l}_{i+}^{(1)}]$. In this case, $\mathcal{C}_{l_{j-}}(v)$, $\mathcal{C}_{l_{j+}}(v)$ could record the minimal value of l_i to keep the same outputs.

Proof of the second fact: Once we have proved the first fact, the second fact is straightforward due to symmetry, i.e., the output of CBLN(l_j, v) would not influence the output of CBLN(l_i, v). □

The practical meaning of Lemma 3 is that, *when we only consider the links' updating impact associated with a local node*, algorithm 2 is resilient to the order of link updates. In conclusion, we successfully tackled Challenge 1.

3.5 Moving from Local to Global Optimality

Before we move to tackle Challenge 2 to explore the global tightest error bound for every link in the constrained *min*-system, we would summarize the sufficient condition where the CBLN(l_i, v) could return the local tightest error bound of $l_i \in \mathcal{L}_v \setminus \mathcal{L}_v^I$ with bounds information of all other links in $\mathcal{L}_{all} \setminus \mathcal{L}_v$ are fixed.

Theorem 2. *Assume that the bounds information of all links in $\mathcal{L}_{all} \setminus \mathcal{L}_v$ are fixed. $\forall l_i \in \mathcal{L}_v \setminus \mathcal{L}_v^I$, whenever $\alpha_v \geq 0$, the CBLN(l_i, v) would return its local tightest error bounds considering the whole min-system and only the fairness and total capacity constraints at node v.*

Note that Theorem 2 is just the summary of our discussion and analysis in the previous section. To recap, when $\alpha_v \geq 0$, we can quickly compute the local tightest error bound by applying one-time CBLN updating to each unidentifiable link. Nevertheless, when $\alpha_v < 0$, we can only solve Problems (3.7.a) and (3.8.a) as a temporary best-effort update. This design in the local CBLN algorithm would help conduct the easier reduction first to the bound of the related unidentifiable link. However, if we want to use CBLN as a building block for exploring the global tightest error bounds of all involved links, we need to investigate how a local CBLN updating at v would influence links bound information in $\mathcal{L}_{all} \setminus \mathcal{L}_v$ and the approaches to handle the influence. All of these will be discussed and analyzed in the following section.

3.6 Tackling Challenge 2: The GES algorithm and The Global Tightest Error Bounds

The CBLN algorithm primarily targets the bound calculation for a single link connected to a node. Note that the definition of the tightest error bound considers all the nodes' correlation constraints. The output of CBLN at node v may influence some CBLN processes of a neighbouring node v'. To explain why, assume that nodes v and v' are connected with a link l. When we update the bound of link l using the constraints associated with node v, this update may substantially influence the updates of other links due to the constraints associated with node v'. This "ping-pong" effect for bound updates needs to be carefully managed to prevent a perpetual cycle of updates over the whole *min*-system, ultimately hindering our ability to reach a conclusion when we want to obtain the tightest error bound for all the involved links.

To avoid the above problem and reduce the redundant calculation at the same node v, we introduce a threshold ϵ and design the ϵ-LUAN(v) algorithm for locally updating all links associated with a node. The main idea is that when the bound improvement of one link is smaller than ϵ, its further changes will not be propagated to other links' bound updates. The details of the ϵ-LUAN algorithm can be found in Algorithm 3.

So far, we have worked out how to update a group of links at a specific node v. We call this step the *node update* step, meaning that all links associated with this node are updated. Next, we further study the inter-impact between node updates, i.e., the cascading impact. To ease discussion, we call a **link** l **stable** at node v if running ϵ-LUAN(v) would not change l's bound. We call a **node** v **stable** if all links associated with v are stable.

It is worth noting that a stable node v may become unstable after when updating its neighbouring node u. This is the main cause of the "ping-pong" effect over the whole *min*-system with correlation constraints. Therefore, we should make as many stable nodes as possible. For this, it would be important to keep track of the returning-unstable nodes after running each ϵ-LUAN. The following procedure helps us to keep track of all unstable nodes due to running ϵ-LUAN(v).

Procedure 1. *In the given min-system with correlation constraints, suppose bounds information of all links in $\mathcal{L}_{all}$ are currently fixed, $\mathcal{N}_v^A$ consists of all the nodes that are connected to v by a current unidentifiable bidirectional link and $v \in \mathcal{N}_{all}$ is a given node.*

Suppose that $v_i \in \mathcal{N}_v^A \cap \mathcal{N}_{all}$ is connected to v by l_a, and we ran ϵ-LUAN(v_i) before. $\forall l_b \in \mathcal{L}_{v_i} \setminus \mathcal{L}_{v_i}^I (l_b \neq l_a)$, $l^-_{v_i(a,b)}, l^+_{v_i(a,b)}$ are stored according to the latest ϵ-LUAN(v_i). And before this tth running ϵ-LUAN(v), $[\hat{l}_{a-}, \hat{l}_{a+}]$ is the current bound of l_a. During this tth ϵ-LUAN(v), we obtain the returned $[\hat{l}'_{a-}, \hat{l}'_{a+}]$ from CBLN(l_a, v).

When $|\hat{l}'_{a-} - \hat{l}_{a-}| + |\hat{l}'_{a+} - \hat{l}_{a+}| \geq \epsilon$, v_i would be collected into $\mathcal{N}^N_{v(t)}$ if it would satisfy at least one of following conditions:

- $\exists l_b \in \mathcal{L}_{v_i} \setminus \mathcal{L}_{v_i}^I$, $\hat{l}'_{a-} > l^-_{v_i(a,b)}$ *or* $\hat{l}'_{a+} < l^+_{v_i(a,b)}$;
- $\exists l_b \in \mathcal{L}_{v_i} \setminus \mathcal{L}_{v_i}^I$, $l_b \in \mathcal{L}^I_{v(t)}$.

$\mathcal{L}^I_{v(t)}$ consists of all the identifiable links caused by this tth ϵ-LUAN(v) and $\mathcal{N}^I_{v(t)}$ collects all the nodes which control at least one link in $\mathcal{L}^I_{v(t)}$. All the nodes that possibly become unstable due to running tth ϵ-LUAN(v) on l_a are collected in $\mathcal{N}^{unstable}_{v(t)} = \mathcal{N}^I_{v(t)} \cup \mathcal{N}^N_{v(t)}$.

Utilizing the above procedure, we develop our GES algorithm(Algorithm 4) to ensure the eventual stabilization of all nodes in $\mathcal{N}_{all}$. To explain the termination of the GES algorithm, we highlight a crucial observation: when a stable node v reverts to an unstable state, it indicates the potential for further reduction in the bounds

of some links connected to this node. This implies that the error bound of a link consistently diminishes over time. Importantly, this reduction process is inherently finite with given ϵ, as the error bound cannot be decreased indefinitely. Consequently, the GES algorithm eventually terminates, guaranteeing the stability of all nodes in the system.

Remark 3. *The GES algorithm is built over the ϵ-LUAN algorithm, which calls the CBLN algorithm. GES is the final algorithm that returns the performance bounds of the network min-system with correlation constraints.*

Complexity Analysis of GES: Given a *min*-system with correlation constraints, in the worst case, $t_\epsilon = 2|\mathcal{L}_{all}| + |\mathcal{N}_{all}| + \sum_{l_i \in \mathcal{L}_{all}} \lceil \frac{\hat{l}_{i+} - \hat{l}_{i-}}{\epsilon} \rceil$ is the maximum number of nodes which require ϵ-LUAN algorithm during the whole GES process. The time complexity of CBLN and ϵ-LUAN is in the same order as solving the convex optimization Problems (3.7) and (3.8) when $\alpha_v \geq 0$ or linear when $\alpha_v < 0$.

Finally, we summarize the property of the GES algorithm with lemma 4.

Lemma 4. *Given a min-system with correlation constraints, assume that $\forall l_i \in \mathcal{L}_{all}, [l_{i-}, l_{i+}]$ denotes its tightest error bound (i.e., the ground-truth). Suppose that $[l^G_{i-}, l^G_{i+}]$ is the GES-returned bound for l_i with certain ϵ,*

- *Correctness: $[l_{i-}, l_{i+}] \subseteq [l^G_{i-}, l^G_{i+}]$;*
- *Conditional Optimality: If $\forall v \in \mathcal{N}_{all}$ equipped with fairness constraint has $\alpha_v \geq 0$, then when $\epsilon \to 0$, $l^G_{i-} \to l_{i-}$ and $l^G_{i+} \to l_{i+}$.*

Proof. The correctness is straightforward because CBLN can guarantee every link's groundtruth value falls within the returned bound.

The conditional optimality is because when $\alpha_v \geq 0$ for all nodes, we can find the local optimal at every node. In addition, GES will block any updates that change a link's bound by ϵ and only allow the updates that reduce a link's current bound. □

3.7 Conclusion

This chapter leverages constraints, whenever available, to improve the bound of a *min*-system. We considered two types of link correlations: fairness constraints and total capacity constraints among a node's adjacent links. We theoretically demonstrated how these correlations can enhance the performance bounds of network *min*-systems.

Our approach systematically addresses two primary challenges in tightening these bounds: the effects of synchronous versus asynchronous updates and the cascading effect. We will evaluate the effectiveness of our solution in Chapter 5.

Algorithm 2: CBLN(l_i, v): Calculating bound for an unidentifiable Link l_i at Node v locally

input : δ_v, b^v_{max}, $\mathcal{L}_v(|\mathcal{L}_v| = m)$, $\mathcal{L}^I_v$ ($|\mathcal{L}^I_v| = n$). For each $l_k \in \mathcal{L}_v \setminus \mathcal{L}^I_v$, its current bound $[\hat{l}_{k-}, \hat{l}_{k+}]$.

output: $\mathcal{C}_{l_{i-}}(v) = \{l^-_{v(j,i)} | l_j \in (\mathcal{L}_v \setminus l_i) \setminus \mathcal{L}^I_v\}$ where $l^-_{v(j,i)}$ is the value that l_j took when $l_i = \hat{l}'_{i-}$ by its (3.7) at v;
$\mathcal{C}_{l_{i+}}(v) = \{l^+_{v(j,i)} | l_j \in (\mathcal{L}_v \setminus l_i) \setminus \mathcal{L}^I_v\}$ where $l^+_{v(j,i)}$ is the value that l_j took when $l_i = \hat{l}'_{i+}$ at v;
The newly obtained bandwidth bound $[\hat{l}'_{i-}, \hat{l}'_{i+}]$

begin
  $\alpha_v = \delta_v m - m + n$;
  if $\alpha_v \geq 0$ **then**
    `/* updates the bounds using both the fairness and total capacity constraints */`
    Obtaining the eligible numerical approximated global optimal solution $\hat{l}'_{i-}$ of (3.7) of l_i at v and formulating corresponding $\mathcal{C}_{l_{i-}}$;
    Obtaining the eligible numerical approximated global optimal solution $\hat{l}'_{i+}$ of (3.8) of l_i at v, formulating corresponding $\mathcal{C}_{l_{i+}}$;
  else
    `/* updates the bounds only using the total capacity constraint */`
    $\hat{l}'_{i-} = \hat{l}_{i-}$;
    $\hat{l}'_{i+} = \hat{l}_{i+}$;
    $l_{new} = b^v_{max} - \sum_{l_k \in \mathcal{L}^I_v} I_k - \sum_{l_k \in (\mathcal{L}_v \setminus l_i) \setminus \mathcal{L}^I_v} \hat{l}_{k-}$;
    if $l_{new} < \hat{l}_{i+}$ **then**
      $\hat{l}'_{i+} = l_{new}$;
      $\mathcal{C}_{l_{i+}}(v) = \{l^+_{v(k,i)} = \hat{l}_{k-} | l_k \in (\mathcal{L}_v \setminus l_i) \setminus \mathcal{L}^I_v\}$;
    end
  end
end

Algorithm 3: ϵ-LUAN(v): Locally Updating links At Node v with tolerance ϵ.

input : ϵ, $\mathcal{L}_v$, $\mathcal{L}_v^I$. For each $l_k \in \mathcal{L}_v \setminus \mathcal{L}_v^I$, $[\hat{l}_{k-}, \hat{l}_{k+}]$

output: For each $l_i \in \mathcal{L}_v \setminus \mathcal{L}_v^I$, updating its bandwidth bound based on the results of CBLN(l_i, v). In addition, according to corresponding $\mathcal{C}_{l_{i-}}(v)$ and $\mathcal{C}_{l_{i+}}(v)$, updating $l^-_{v(j,i)}$ and $l^+_{v(j,i)}$ at each $l_j \in (\mathcal{L}_v \setminus l_i) \setminus \mathcal{L}_v^I$.

begin
  Adding all links in $\mathcal{L}_v \setminus \mathcal{L}_v^I$ to the list L;
  while L *is not empty* **do**
    $l_i \leftarrow L.pop(0)$;
    Running CBLN(l_i, v) and obtaining $[\hat{l}'_{i-}, \hat{l}'_{i+}]$, $\mathcal{C}_{i-}(v)$, $\mathcal{C}_{i+}(v)$;
    if $|\hat{l}'_{i-} - \hat{l}_{i-}| + |\hat{l}'_{i+} - \hat{l}_{i+}| \geq \epsilon$ **then**
      Updating current bound of l_i to $[\hat{l}'_{i-}, \hat{l}'_{i+}]$;
      for $l_j \in (\mathcal{L}_v \setminus l_i) \setminus \mathcal{L}_v^I$ **do**
        Updating $l^-_{v(j,i)}$, $l^+_{v(j,i)}$ based on $\mathcal{C}_{i-}(v)$, $\mathcal{C}_{i+}(v)$;
      end
      if l_i *becomes identifiable* **then**
        Adding l_i to $\mathcal{L}_v^I$;
      end
    end
    for $l_k \in L$ **do**
      if $\hat{l}_{k-} == \hat{l}_{i-}$ *and* $\hat{l}_{k+} == \hat{l}_{i+}$ *and* $|\hat{l}'_{i-} - \hat{l}_{k-}| + |\hat{l}'_{i+} - \hat{l}_{k+}| \geq \epsilon$ **then**
        Removing l_k from L;
        Updating current bound of l_k to $[\hat{l}'_{i-}, \hat{l}'_{i+}]$;
        $l^-_{v(i,k)} = l^-_{v(k,i)}$;$l^+_{v(i,k)} = l^+_{v(k,i)}$;
        for $l_j \in (\mathcal{L}_v \setminus (l_k \cup l_i)) \setminus \mathcal{L}_v^I$ **do**
          $l^-_{v(j,k)} = l^-_{v(j,i)}$; $l^+_{v(j,k)} = l^+_{v(j,i)}$;
        end
        if l_k *becomes identifiable* **then**
          Adding l_k to $\mathcal{L}_v^I$;
        end
      end
    end
  end
end

Algorithm 4: GES: Global ϵ-stabilizing algorithm on $\mathcal{N}_{all}$

input : $\mathcal{N}_{all}$

output: The performance bounds of all links in $\mathcal{L}_{all}$.

begin
  Initialize updating queue $\mathcal{Q}_{update}$ and add all nodes in $\mathcal{N}_{all}$ to $\mathcal{Q}_{update}$;
  while $\mathcal{Q}_{update}$ *is not empty* **do**
    $v_i \leftarrow \mathcal{Q}_{update}.pop(0)$;
    Running ϵ-LUAN(v_i);
    According to Procedure 1, obtain corresponding $\mathcal{N}_{v_i(t)}^{unstable}$;
    Add all nodes in $\mathcal{N}_{v_i(t)}^{unstable}$ to $\mathcal{Q}_{update}$.
  end
end

Chapter 4

A Reinforcement Learning Approach for Path Construction

In previous chapters, we assumed that the measurement paths (MPs) are given and derived the performance bounds under this assumption. In this chapter, we investigate how to build MPs step by step, based on which we can derive error bounds close to the global tightest error bounds. Constructing MPs sequentially in our context is similar to playing a chess game: once an MP is built and its end-to-end bandwidth is probed, we cannot regret if the MP does not help reduce the error bounds, since the cost involved in the measurement has occurred. Therefore, we borrow the similar idea in developing a chess game and use the special knowledge in bandwidth tomography as well as off-policy[1] reinforcement learning for constructing MPs.

4.1 On the Hardness of Path Construction for the (Global) Tightest Error Bound

In Chapter 2, the tightest error bound of every link is conditional on a given *min*-system. The tightest error bound is called the *global* tightest error bound if it is the smallest among all possible error bounds derived for the link with different sets of MPs. Obviously, the error bound derived with all possible MPs is the global tightest error bound. Nevertheless, the total number of possible MPs may be huge, and it is well known that listing MPs between two monitors is $\#P$-complete [38]. We hence

[1]Off-policy means that learning is from data "off" the target policy, i.e., the policy being learned about [53].

need to answer two questions: (1) Is it possible to find the error bound identical to the global tightest error bound without listing all possible MPs? (2) Can we design a method to reduce the number of MPs to achieve the error bound close to the *global* tightest error bound? In the rest of this chapter, the tightest error bound by default means the global tightest error bound unless stated otherwise.

First of all, we *only need to study the bandwidth tomography with two monitors*. This is because if there are multiple monitors, we can introduce two virtual monitors such that a virtual monitor only has virtual links of bandwidth b_{max} to connect each (physical) monitors. Then the multi-monitor bandwidth tomography problem is reduced to bandwidth tomography problem with the two virtual monitors. The concept of virtual monitors was also used in [37] to simplify theoretical analysis.

In this section, we show the *negative* answer to the first question. For this, we only need to construct a scenario and prove in this scenario that we must list all possible MPs between the two monitors in order to find the tightest error bound for every link. The proof needs two preliminary results: Lemma 5 and Lemma 6.

Lemma 5. *For a given network $\mathcal{G}$ and two monitors, assume that $\mathcal{P}_m$ is a set of MPs that covers all links in $\mathcal{G}$. With CTB, we can obtain d nonempty sets $\{S_{b'_j}\}, j = 1, 2, \ldots, d$ and the corresponding identifiable links from $\mathcal{P}_m$. When a new MP P_{m+1} is added, we denote its end-to-end bandwidth as b_{m+1} and the set of MPs as $\mathcal{P}_{m+1}$. Let $b'_{\min}, b'_{\max}$ denote the minimum and maximum $b'_j (j = 1, 2, \ldots, d)$ whose $S_{b'_j}$ contains at least one link on P_{m+1}, respectively. Denote $S_{update} = \{l | l \in P_{m+1}, l$ is not identifiable in $\mathcal{P}_m\}$, i.e., a set of links whose tightest error bounds may be updated due to the addition of P_{m+1}.*

If $|S_{update}| > 0$, we need to update the d nonempty sets based on the value of b_{m+1}:

- ***Case a:*** *If $b_{m+1} \notin \{b'_j, j = 1, 2, \ldots, d\}$:*
  - ☐ ***Case a1:*** *If $b_{m+1} > b'_{\max}$, we move all the links in S_{update} from their original sets to a new nonempty set $S_{b_{m+1}}$.*
  - ☐ ***Case a2:*** *If $b_{m+1} < b'_{\max}$, let b'_{j_m} denote the minimum $b'_j (j = 1, 2, ..., d)$ that is strictly larger than b_{m+1}. Denote $S_{remained} = \bigcup_{b'_j \geq b'_{j_m}} S_{b'_j}$. If $|S_{update} \setminus S_{remained}| > 0$, we move all the links in $S_{update} \setminus S_{remained}$ from their original sets to a new nonempty set $S_{b_{m+1}}$.*
- ***Case b:*** *If $b_{m+1} \in \{b'_j, j = 1, 2, \ldots, d\}$:*

- ☐ ***Case b1:*** *If* $|S_{update} \cap S_{b'_{\min}}| = 1$*, the link in* $S_{update} \cap S_{b'_{\min}}$ *is identifiable.*
- ☐ ***Case b2:*** *If* $|S_{update} \cap S_{b'_{\min}}| > 1$*, we denote* $S_{remained} = \bigcup_{b'_j \geq b_{m+1}} S_{b'_j}$*. If* $|S_{update} \setminus S_{remained}| > 0$*, we move all the links in* $S_{update} \setminus S_{remained}$ *from their original sets to* $S_{b_{m+1}}$*.*

After the update, if any updated set has only one link, this link is identifiable. The updated nonempty sets and the new identifiable links are the same as those obtained from $\mathcal{P}_{m+1}$ *with* CTB*.*

Proof. For a given *min*-system, we have shown in Section 3.3 that CTB can not only determine the tightest lower bound for the unidentifiable links but also find out all identifiable links. When the new path P_{m+1} is introduced to the *min*-system with end-to-end bandwidth b_{m+1}, we should arrange all the links in P_{m+1} to their suitable sets and find out all identifiable links in the new *min*-system $\mathcal{P}_{m+1}$ to make sure the updated sets are same as those obtained from $\mathcal{P}_{m+1}$ with CTB.

To ease discussion, we first note the following two facts:

1) All links whose bounds need to be updated (i.e., moved from one set to another set) are contained in $S_{updated}$. After the update, some links in $\mathcal{G}$ may become identifiable.

2) $b_{m+1} \geq \min_{j(j=1,2,3,\ldots,d)} b'_j$;

The first fact is straightforward based on the definition of $S_{updated}$. The second fact is obvious because $\mathcal{P}_m$ covers all links in $\mathcal{G}$. If $|S_{updated}| = 0$, P_{m+1} would not bring any new information to the original *min*-system because all links on P_{m+1} are identifiable already. So we only need to consider the cases listed in Lemma 5:

- **Case a:** In this case, b_{m+1} is different from all the existing end-to-end bandwidth in $\mathcal{P}_m$.
  - ☐ **Case a1:** If $b_{m+1} > b'_{\max}$, all links in $S_{updated}$ would have larger lower bounds. A new set S_{m+1} would be generated to contain them.
  - ☐ **Case a2:** If $b_{m+1} < b'_{\max}$, some links in $S_{updated}$, whose lower bandwidth bound obtained from $\mathcal{P}_m$ are larger than b_{m+1}, do not need to be updated. These links are all included in $S_{remained}$. If $|S_{updated} \setminus S_{remained}| > 0$, a new set $S_{b_{m+1}}$ would be generated to carry the links in $S_{updated} \setminus S_{remained}$.

- **Case b:** In this case, b_{m+1} has already existed in end-to-end bandwidth in $\mathcal{P}_m$. Note that $b'_{\min} \leq b_{m+1} \leq b'_{\max}$.

  - ☐ **Case b1:** Due to Theorem 1, the link in $S_{updated} \cap S_{b'_{\min}}$ is identifiable.
  - ☐ **Case b2:** If $|S_{updated} \cap S_{b'_{\min}}| = 0$, no update is needed. If $|S_{updated} \cap S_{b'_{\min}}| > 1$, some links whose lower bandwidth bound obtained from $\mathcal{P}_m$ are equal or larger than b_{m+1} in $S_{updated}$ do not need to be updated. These links are all included in $S_{remained}$. If $|S_{updated} \setminus S_{remained}| > 0$, we need to move the links in $S_{updated} \setminus S_{remained}$ from their original sets to $S_{b_{m+1}}$.

  Due to the movement of links, some updated sets may have only one link, and in this case this link is identifiable.

☐

Lemma 5 indicates that we can perform sequential update on the (conditional) tightest error bounds when a new MP is constructed. Lemma 5 is essentially another way to perform "variable elimination" (Line 3-Line 10 of CTB) on $\mathcal{P}_{m+1}$.

Lemma 6. *For a given network $\mathcal{G}$ and two monitors, assume that $\mathcal{P}_m$ is a set of MPs. With CTB, we can obtain a group of d nonempty sets $\{S_{b'_j}\}, j = 1, 2, \ldots, d$ and the identifiable links from $\mathcal{P}_m$. $\mathcal{P}_m$ can obtain the greatest lower bound of each link in $\mathcal{G}$ if and only if it satisfies the following three conditions:*

1) $\mathcal{P}_m$ covers all links in $\mathcal{G}$;

2) For any other d' nonempty sets derived from a different set of paths $\mathcal{P}'_{m'}$ by CTB, $d \geq d'$;

3) Any new MP P_{m+1} will not cause the movement of link(s) between two distinct nonempty sets by Lemma 5.

Proof. Lemma 6 is straightforward based on lemma 5. ☐

Lemma 6 gives us criteria to determine whether a set of paths $\mathcal{P}_m$ can obtain the tightest error bound of each link in the network $\mathcal{G}$. Nevertheless, in the context of bandwidth tomography, the bandwidth of each link is unknown before hand. As such, the criteria only serves as a guideline. It does not warrant a polynomial-time solution to the first question raised in this section. With construction (using the concept of graph cut in particular), we can show that there is a class of special cases in which we have to probe all possible paths to obtain the tightest error bounds.

Theorem 3. *In the worst case, it is impossible to derive the error bound identical to the global tightest error bound without listing all possible MPs.*

Proof. We only need to construct a case where we must list all possible MPs to derive the global tightest error bounds.

Given two monitors, assume that the length of any MP is at least two (because otherwise the link bandwidth of the MP is immediately available). Let S_{cut} denote a set of links forming a *cut* of $\mathcal{G}$. Assume that the two monitors are separated into two components by S_{cut}. Assume that all links in S_{cut} have the same bandwidth b_{cut}. Assume that the bandwidth of other links in $\mathcal{G}$ is *at least* b_{cut}. We have the following two facts:

1) Every link in $\mathcal{G}$ is not identifiable;

2) Sequentially probing the MPs in $\mathcal{G}$, we have to test all the possible MPs to obtain the tightest bounds of each link in $\mathcal{G}$.

To prove 1), any MP must have one link in S_{cut} because S_{cut} is a cut set. Besides, the bandwidth of each link in $\mathcal{G}$ is at least b_{cut}. Hence, all MPs between the two monitors must have the same end-to-end bandwidth b_{cut}. According to Theorem 1 none of links is identifiable.

To prove 2), the end-to-end bandwidth of any new path P_{new} is b_{cut}. Based on Lemma 5, no new path could improve the highest lower bound of each link in $\mathcal{G}$. However, for sequentially probing the possible MP in $\mathcal{G}$, we do not know this fact before hand. If we have probed a set of paths $\mathcal{P}_m$, all we know is that all links on the probed paths have the same conditional highest lower bound b_{cut}. They are all contained in the same set $S_{b_{cut}}$. No additional information could help to choose the new MP. Any new MP would have the same chance to reduce the error bound of link. Therefore, according to Lemma 6, we have to test all possible MPs to make sure that there is no new path P_{new} to raise the lower bound of each link in $\mathcal{G}$. □

Theorem 3 means that finding the global tightest error bound for links is $\#P$-complete [38]. To tackle the challenge, we adopt a reinforcement learning approach that utilizes the special knowledge in bandwidth tomography for effective learning.

4.2 Special Knowledge in Bandwidth Tomography for Action Design

In Section 4.1, we showed that we only need to analyze the case of two (virtual) monitors to ease theoretical analysis. This section does not need the concept of virtual monitors, and all monitors are referred as physical monitors.

Based on Lemma 5 and Lemma 6, it is easy to have the following proposition:

Proposition 1. *For a set of MPs $\mathcal{P}$, the sufficient and necessary conditions for it to give the global tightest error bounds are:*

i) Every link in the network $\mathcal{G}$ must be contained in at least one MP in $\mathcal{P}$, i.e., $\mathcal{P}$ covers the whole network.

ii) Every identifiable link l_I must be identified by the min-system formed by $\mathcal{P}$.

iii) For any link l_U that is unidentifiable, if there exists an MP P such that 1) l_U is contained in P and 2) every other link on P has bandwidth no smaller than that of l_U, P must be included in $\mathcal{P}$.

Proposition 1 gives insights on how to take proper actions in a reinforcement learning-based MP construction method.

Analysis for identifiable links: Based on Proposition 1, we can see that to obtain the global tightest error bounds, the constructed MPs should be able to identify a link l if l is identifiable. We first give a sufficient condition for identifying identifiable links in the following lemma.

Lemma 7. *A sufficient condition to identify an identifiable link l_I is that there is at least one pair of MPs (P_1, P_2) satisfying the following conditions:*

i) (I-structure): for every link l in P_1 except l_I, $l \in P_2$ (as illustrated in Fig. 4.1).

ii) The measurement value of P_1 is smaller than that of P_2.

Proof. Given an I-structure, assume that P_1's measurement value is b_{P_1} and P_2's measurement value is b_{P_2}. From the definition of the I-structure we know that every link l in P_1 except the link l_I has a bandwidth value higher than b_{P_1} because $b_{P_1} < b_{P_2}$. Therefore, the bandwidth value of l_I must be exactly equal to b_{P_1}, i.e., l_I is identifiable. □

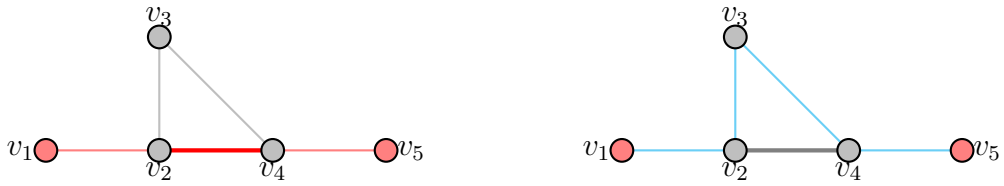


Figure 4.1: Illustration of *I-structure*: P_1 and P_2 are marked in red and cyan, respectively; every link of P_1 except $l_{2,4}$ is also a link of P_2.

Note that knowing the sufficient condition for identifiability is not equivalent to having a polynomial-solution to determine whether a given link is identifiable or not. For a given link l, we do not know in advance which MPs will satisfy the conditions in Lemma 7, since the measurement value of an MP can only be revealed after a trail (sending probes along the MP). However, we are able to select MPs that satisfy the topological requirement (i), i.e., when we select a new MP P_{new} into the measurement path set $\mathcal{P}$, we could try to select a P_{new} that forms an *I-Structure* with an MP in $\mathcal{P}$. This knowledge helps to (gradually) build the appropriate $\mathcal{P}$. More details on the action based on the above analysis are described in Section 4.3.2.

Analysis for unidentifiable links: For an unidentifiable link l_U, even if we include all possible MPs in $\mathcal{G}$, l_U still cannot be identified, and the possibly tightest error bound of l_U is given by the following lemma.

Lemma 8. *The possibly tightest error bound for an unidentifiable link l_U is $b_{max} - b_{l_U}$, where b_{l_U} is the bandwidth value of l_U and b_{max} is the maximum bandwidth over all links in the network.*

Proof. For an unidentifiable link l_U, even if we list all the possible MPs, we cannot reduce its error bound to 0. Because we can only infer l_U's bandwidth from MPs containing l_U and every MP containing l_U must have a measurement value smaller or equal to b_{l_U}, the possibly tightest error bound of l_U is $b_{max} - b_{l_U}$. □

Based on Lemma 8, to achieve the smallest error bound for an unidentifiable link l_U, we need to find an MP P that satisfies 1) P includes l_U, and 2) every other link on P has bandwidth no smaller than b_{l_U}. In other words, we wish the bandwidth of every other link along the path to be as large as possible, so that b_{l_U} may become the smallest and thus the measurement value of P. In this way, l_U can achieve its

smallest error bound. Based on this analysis, we design the corresponding actions in Section 4.3.2.

4.3 Guided Sequential Path Construction (GSPC)

4.3.1 Overview

In reinforcement learning (RL), an *agent* takes *actions* and interacts with the *environment*, which gives the agent feedback for the action in form of *reward* [53]. The goal of the agent is to maximize the cumulative total reward in the long run. An RL problem is usually cast in the framework of Markov Decision Process (MDP) [53], where the agent can make action decisions round-by-round based on the current state of the environment. In the context of constructing MPs for bandwidth tomography, the core RL elements are:

- **Environment**: The environment consists of the network topology $\mathcal{G}$ and its (hidden) bandwidth value b_i on each link l_i.
- **Agent**: It is a controller that decides which MP to traverse at each state.
- **Action**: An action a_t of the agent means constructing an MP and measuring the bandwidth of the MP.
- **Policy**: It is a mapping from the agent's perceived states of the environment to the actions to be taken when in those states.
- **State**: A state is defined as $s_t = \mathcal{P}_t$, where $\mathcal{P}_t$ denotes the set of constructed MPs up to round t.
- **Reward**: The reward for taking action a_t at state s_t is the *negative* total error bound computed with CTB over $\mathcal{P}_{t+1}$.

We adopt a model-free off-policy RL approach, a variant of Q-learning, and use Guided Sequential Path Construction (*GSPC*) to guide the agent to make better decisions. Off-policy means that the agent is trained with offline simulated networks at *each step*. The simulated networks have the *same* topology of the target network, i.e., the network for which we need to construct MPs. As time rolls out, we can either identify or derive the bounds of links in the target network with existing MPs built so

Algorithm 5: OfflineTrainer

input : current state in target network s_t, training rounds N, target network $\mathcal{G}$

output: action a_t suggested by offline trainer

get each covered link's bound with Algorithm CTB;
determine phase 1 or phase 2, and according to the phase initialize reward $R_i = 0$ $(i = 1, 2, 3)$ for the actions $a_i (i = 1, 2, 3)$, respectively, in the action space;
for *round* $j \leftarrow 1$ **to** N **do**
    initialize a simulated network $\mathcal{SN}_j$ that has the same topology as $\mathcal{G}$ but has no bandwidth value assigned to each link;
    foreach *link* l *in* $\mathcal{SN}_j$ **do**
        if l *in* s_t **then** select a random number r between l's bound interval (calculated with CTB) $[l_{lower\_bound}, l_{upper\_bound}]$;
        /* l_{lower_bound} and l_{upper_bound} for an identified link are the same */
        else choose a random number $r \in [0, b_{max}]$;
        assign r as l's bandwidth value in $\mathcal{SN}_j$;
    end
    foreach $a_i (i = 1, 2, 3)$ **do**
        build a new MP P_{new} by performing action a_i on simulated network $\mathcal{SN}_j$;
        $s'_t = s_t + P_{new}$; /* add the new MP to the set s_t */
        get the total error bound *teb* under s'_t with CTB;
        $R_i = R_i + teb$;
    end
end
return $a_i = argmax_i(-R_i)$;

far. For each identified link in the target network, we set the same bandwidth value for the corresponding link in each simulated network; for each link where we only know the bounds in the target network, we set a random number uniformly distributed within the lower and upper bounds for the corresponding link in each simulated network. We use uniform sampling based on the Principle of Indifference [54], meaning that in the absence of relevant information, the best way to assign probabilities is to distribute them evenly across all possible outcomes.

The rationale of this offline training is that if the policy is learned from many simulated networks of the same topology, the policy should (statistically) work well for the target network as well. After this offline training, the agent updates the policy, i.e., action-value function $Q(s_t, a_t)$, iteratively to quantify the predicted quality of taking action a_t at state s_t. The process repeats until the designated number of MPs is reached.

One special difficulty is to control the dimension of the action space, because the general term of action "generating a new path" would result in an exponential number of actions. Fortunately, the special knowledge introduced in the previous section can help us design the appropriate action space.

4.3.2 Policy

The policy considers the two phases of the network. First, we need to cover all links, because if a link has not been covered, no information regarding this link is available. Hence, our first goal in building MP is to cover all links (Phase 1). After that, we build more paths to further lower down error bounds (Phase 2).

Phase 1: before the network $\mathcal{G}$ is covered by the selected MPs (i.e., not every link of $\mathcal{G}$ is included in at least one MP):

- *Action-Random-b* (AR_b): randomly select an MP between two random monitors, and guarantee that at least one uncovered link is contained in this MP.
- *Action-I-b* (AI_b): utilize the smallest MP P_{smp}, i.e., MP with the smallest measurement value in $\mathcal{P}_t$, and select a new MP P_{new} which contains at least one uncovered link and forms an *I-Structure* with P_{smp}.
- *Action-U-b* (AU_b): utilize the biggest MP P_{bmp}, i.e., MP with the largest measurement value in $\mathcal{P}_t$, and take a random inner vertex of P_{bmp} v_r such that $P_{bmp} = M_1 \to v_r \to M_2$, $P_{new} = M_1 \to v_r \to M_3$, where M_3 is a third monitor,

and the segment $v_r \rightarrow M_3$ is node-disjoint with P_{bmp}. In addition, P_{new} contains at least one uncovered link.

Phase 2: after $\mathcal{G}$ is covered by selected MPs:

- *Action-Random-a* (AR_a): randomly select an MP between two random monitors.
- *Action-I-a* (AI_a): utilize the smallest MP P_{smp}, and select a new MP P_{new} that forms an *I-Structure* with P_{smp}.
- *Action-U-a* (AU_a): utilize the biggest MP P_{bmp}, and take a random inner vertex of P_{bmp} v_r such that $P_{bmp} = M_1 \rightarrow v_r \rightarrow M_2$, $P_{new} = M_1 \rightarrow v_r \rightarrow M_3$, where M_3 is a third monitor, and the segment $v_r \rightarrow M_3$ is node-disjoint with P_{bmp}.

Note that the subscripts b and a in the actions denote different phases, i.e., before and after all links are covered by the constructed MPs.

Remark 4. *The actions utilize the knowledge of bandwidth tomography. In particular, the actions of AI_b and AI_a target at covering identifiable links (based on Lemma 7), and the actions of AU_b and AU_a target at covering unidentifiable links (based on Lemma 8). The random actions (AR_b and AR_a) give the agent a chance of exploring other possibilities besides the guided searches. In Phase 1, each action needs to cover at least a new link as the basic requirement. This guarantees that MPs will eventually cover $\mathcal{G}$, instead of hovering over covered links for a long time. Once all links are covered, we do not need to consider this requirement in Phase 2.*

The pseudo code of the offline training and *GSPC* is shown in Algorithm 5 and Algorithm 6, respectively.

Algorithm 6: Guided Sequential Path Construction (*GSPC*)

input : $\mathcal{G}$, total number of MPs T, offline training rounds N
output: T measurement paths

Initialize state $s = Null$;
for *episode* $t \leftarrow 1$ **to** T **do**
    observe state s;
    $a_t = \text{OfflineTrainer}(s, N)$;
    get an MP P by performing action a_t on the target network $\mathcal{G}$;
    $s = s + P$; `/* add the new MP to set` s `*/`
end
return *s;*

Chapter 5

Performance Evaluation

In this chapter, we evaluate our solutions for bandwidth tomography with extensive simulations. Note that the CTB algorithm proposed in Chapter 2 serves as the basis to calculate the tightest error bounds once measurement paths are known. Since CTB has been theoretically proven to return the tightest error bounds in polynomial time, we only need to focus on the effectiveness of our measurement path construction method *GSPC*. In addition, the GES algorithm proposed in Chapter 3 is based on the assumption that we have access to extra information. The main purpose of Chapter 3 is to leverage the extra information as constraints to tighten the bounds of CTB further. As such, we also evaluate the effectiveness of GES in enhancing performance bounds.

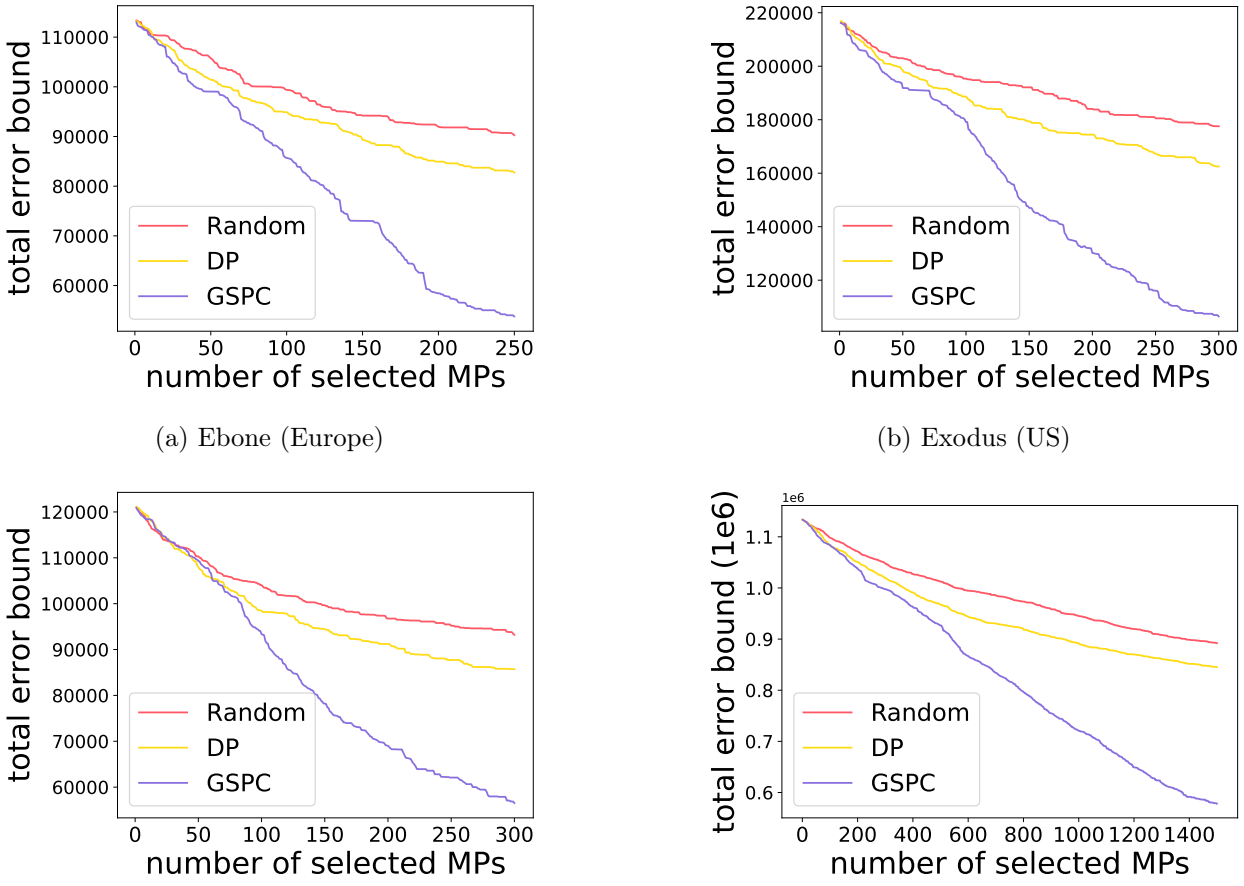
5.1 Evaluation of GSPC

Since there is no existing work to address the path construction problem for bandwidth tomography, we compare *GSPC* with two naïve methods:

- *Random*: Randomly generate an MP at each round.
- *Diversity Preferred (DP)*: Before the graph is covered, select an MP that consists of at least one uncovered links; after the graph is covered, use *Random* for generating new MPs.

We use the metric total error bound (TEB), defined as the sum of error bounds of all links in the network, to show the advantages of *GSPC* over the above two methods. We evaluate their performance with real-world ISP networks (Section 5.1.1). While we

do not exclude the possibility that other better path construction methods might be found in the future, our evaluation results in small-scale simulated networks, where the ground-truth global optimal solutions can be numerically calculated (Section 5.1.2), suggest that the room for further improving *GSPC* might be marginal.



(a) Ebone (Europe)

(b) Exodus (US)

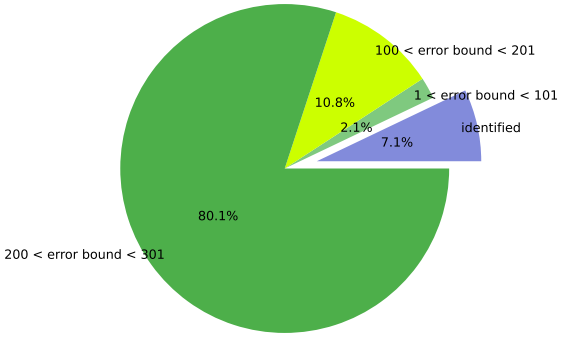
(c) Tiscali (Europe)

(d) Sprintlink (US)

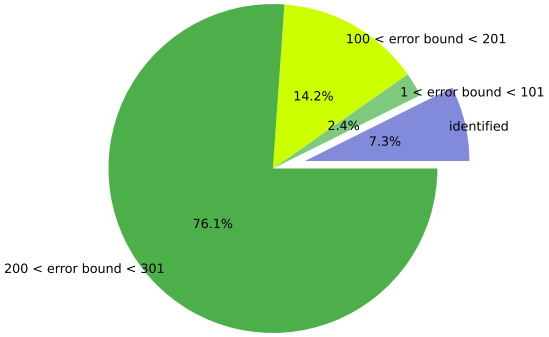
Figure 5.1: Performance of random, diversity preferred and GSPC.

5.1.1 Experiment on Real-World ISP Networks

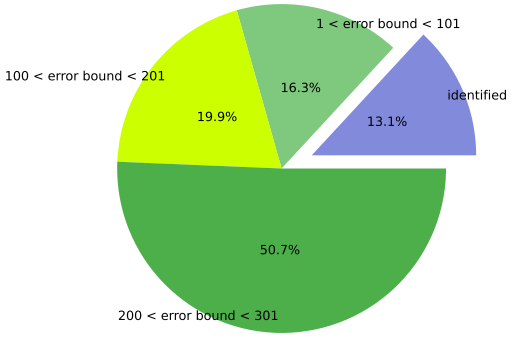
We select four real-world autonomous system (AS) networks collected by the Rocketfuel project [55]. The networks have different sizes, whose parameters are listed in Table 5.1. In each network, the ground-truth bandwidth of each link is set to a random integer in $[2, 300]$ for Ebone and Tiscali, and a random integer in $[2, 500]$ for Exodus and Sprintlink.



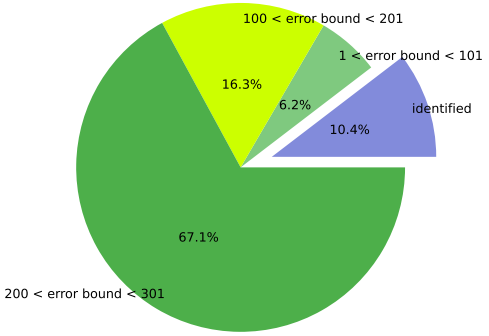
(a) Ebone Random



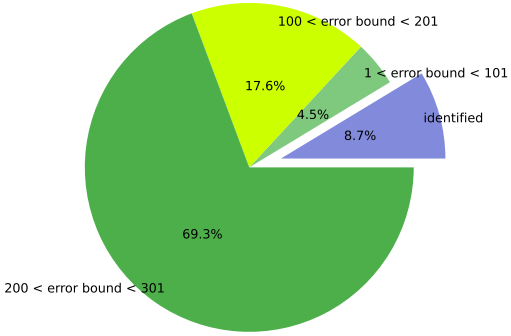
(b) Ebone DP



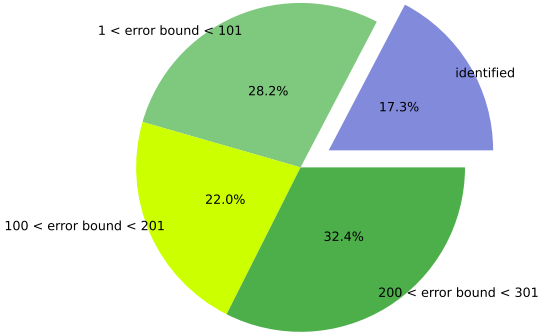
(c) Ebone GSPC



(c) Tiscali Random



(d) Tiscali DP



(e) Tiscali GSPC

Figure 5.2: The distribution of error bounds.

Table 5.1: Network parameters

ISP name	$\|L\|$	$\|V\|$	Average node degree
Ebone (AS1755)	381	172	4.43
Exodus (AS3967)	434	201	4.31
Tiscali (AS3257)	404	240	3.36
Sprintlink (AS1239)	2268	604	7.51

Table 5.2: Ratio of the number of identified links over the total number of links

ISP name	Random	DP	GSPC
Ebone (AS1755)	27/381	28/381	50/381
Exodus (AS3967)	23/434	27/434	64/434
Tiscali (AS3257)	42/404	35/404	70/404
Sprintlink (AS1239)	194/2268	210/2268	411/2268

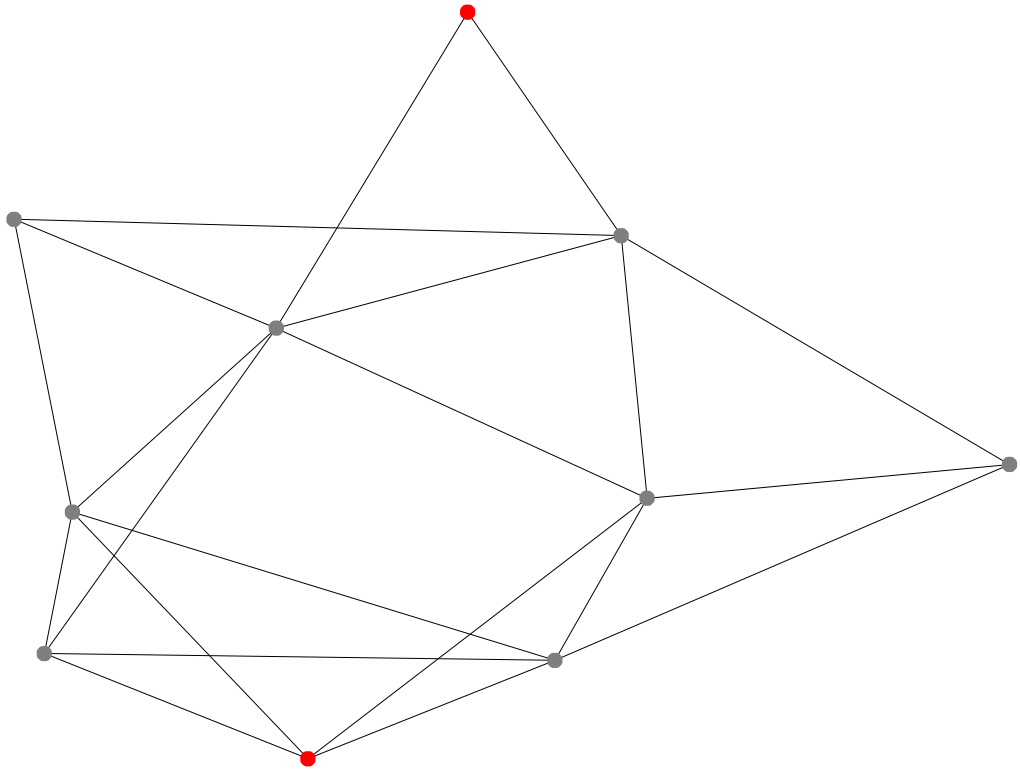
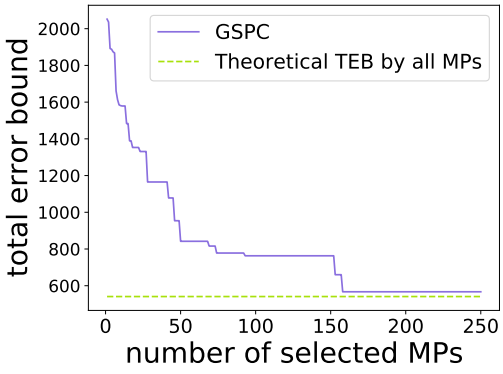


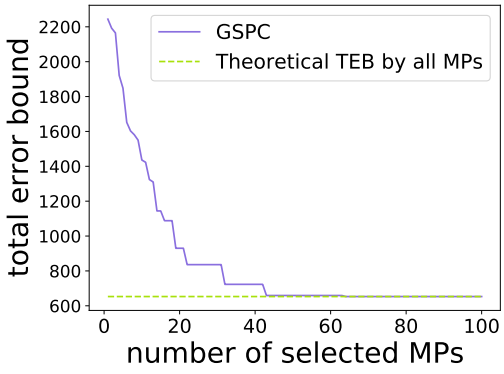
Figure 5.3: An example small network where we can list all possible MPs (2 monitors are in red color).

In order to make sure that the graph can be covered by simple MPs, every dangling points (i.e., node with degree 1) has to be a monitor and each bi-connected components must have at least 2 monitors inside. Besides the above two necessary conditions for covering the whole network, a small random number of monitors are deployed based on the size of each network. For each of the ISP network, we conduct the experiment for the three different methods until designated number of MPs is exhausted (250 for Ebone, 300 for Exodus and Tiscali, 1500 for Sprintlink). These numbers of MPs are chosen based on the observation that the numbers are high enough to cover all the links in the network with both *DP* and *GSPC*.

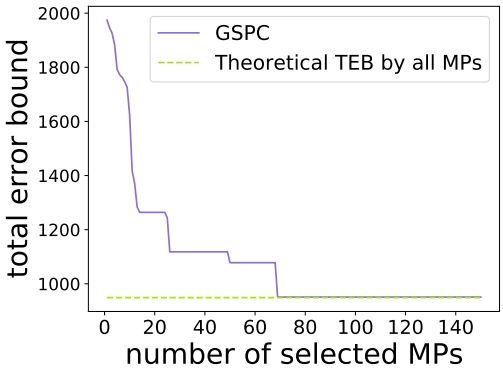
We perform multiple runs, which all show similar performance trends. To save space, we only show the performance result of one sample run in Fig. 5.1. We can see that with the increase in number of MPs, the advantage of *GSPC* becomes significant. Compared to *Random* and *DP*, the TEB of *GSPC* is decreased with the fastest speed, regardless of the topology. *DP* outperforms *Random* except in the beginning phase.



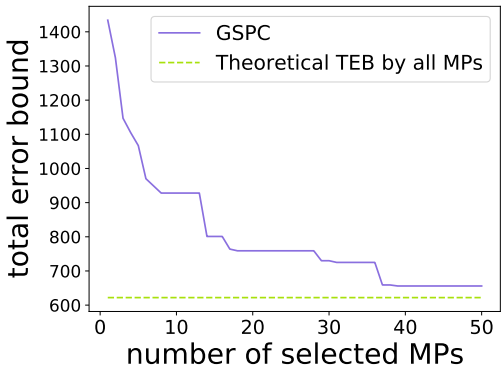
(a) Random network I (312 MPs in total)



(b) Random network II (695 MPs in total)

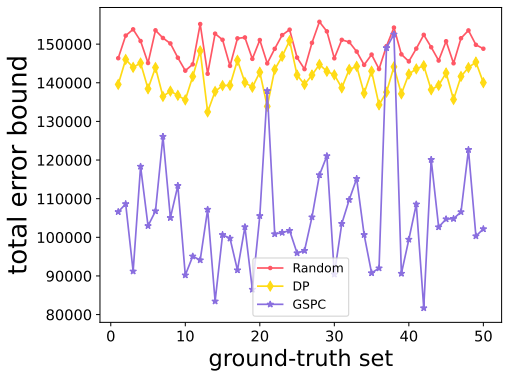


(c) Random network III (252 MPs in total)

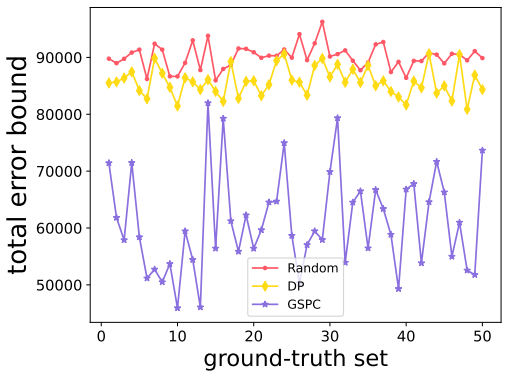


(d) Random network IV (85 MPs in total)

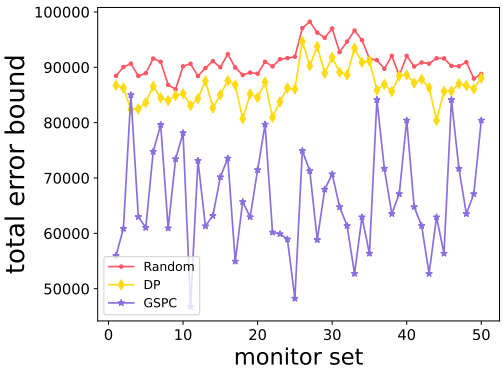
Figure 5.4: Compare the performance of *GSPC* with theoretical smallest TEB (TS-TEB).



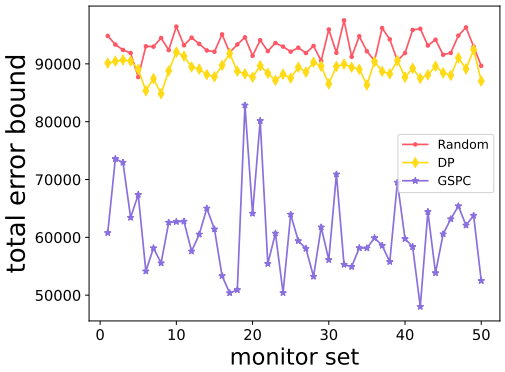
(a) Ebone
(Avg: Random=149150.32, DP=141099.86
GSPC=105207.10)



(b) Tiscali
(Avg: Random=91163.40, DP=86550.38
GSPC=66331.10)



(c) Ebone
(Avg: Random=90107.4, DP=85740.74
GSPC=60987.28)



(d) Tiscali
(Avg: Random=93164.2, DP=88939.98
GSPC=60647.52)

Figure 5.5: Stability experiment: (a) (b) results with different ground-truth sets, and (c) (d) results with different monitor sets.

Computed with the MP set at the first round and at the last round, the average TEB reductions of *Random*, *DP* and *GSPC* are 82827, 102077, 197639, respectively. *GSPC* brings a 238% improvement over *Random* and 193% improvement over *DP*.

Besides evaluating the total error bound (TEB), we also calculate the ratio of number of identified links over the total number of links with Random, DP, and GSPC, respectively. Table 5.2 shows the results. We can see that GSPC always achieve much more identified links than Random and DP. To have a clearer view on the error bounds of unidentified links, we draw pie charts in Fig. 5.2 to display the distribution of error bounds of Ebone and Tiscali. Exodus and Sprintlink have similar pie charts and are omitted to save space. The results show that the error bounds with Random and DP are heavily clustered in the high values (i.e., between 200 and 301). In summary, GSPC fulfills two-fold superiority over Random and DP: identifying more links and giving smaller error bounds for unidentified links.

5.1.2 Performance of *GSPC* v.s. Theoretical Smallest TEB (TS-TEB)

From the previous experiment results, *GSPC* has significantly better performance than the two baseline methods. Nevertheless, due to the large network size, we cannot afford finding the theoretical smallest TEB (TS-TEB), i.e., TEB calculated with the set of all possible MPs, and thus we still do not know how close the performance of *GSPC* is to TS-TEB. For this study, we create 4 randomly-generated small networks where we can practically list all MPs. One example small network is shown in Fig. 5.3. For each network, we assign each link's bandwidth value a random value in $[2, 100]$ and randomly select 2 nodes as monitors. We generate all simple paths between the monitors, based on which the TS-TEB is computed. We then conduct *GSPC* and compare its performance with TS-TEB. We can see that in two networks (Figs. 5.4 (b) and (c)) *GSPC* achieves TS-TEB with a much smaller number of MPs (the total number of all possible MPs listed in the captions). In addition, the gap between the TEB with *GSPC* and TS-TEB is very small in the other two networks (Figs. 5.4 (a) and (d)).

5.1.3 The Stability of GSPC

Furthermore, we conduct a series of experiments to test the stability of GSPC in two aspects:

1. stability with respect to different sets of ground-truth.

2. stability with respect to different monitor placements.

Using the variable-control method, we firstly fix the monitor placement and test how different ground-truth sets affect the performance of *GSPC*. We have observed similar phenomena on all the four real-world ISP networks and only show the results in two networks (Ebone and Tiscali) to save space. In each network, we randomly select a monitor placement that fulfills the monitor selection requirement in Section 5.1.1. After that, we perform 50 rounds of experiment with different random ground-truth sets. In each round, *Random*, *DP*, and *GSPC* select the designated number of MPs (250 for Ebone and 300 for Tiscali). The results in Figs. 5.5 (a) and (b) shows the TEBs achieved by *Random*, *DP*, and *GSPC* in each run. Similarly, to test the above objective 2), we control ground-truth as an invariant factor and use 50 different monitor placements (i.e., 50 runs). For each monitor placement, after the designated number of MPs are exhausted (250 for Ebone and 300 for Tiscali), we record the corresponding TEBs obtained with each method. The results of each run are shown in Figs. 5.5 (c) and (d).

The testing results in Fig. 5.5 clearly show that *GSPC* nearly always outperforms *Random* and *DP*, no matter how we change the ground-truth values and the monitor placements. Among the 50 runs, *GSPC* is always better than *Random* and *DP* in the Tiscali network, and is worse than *DP* only in one or two exceptions in Ebone. After a closer look at the exception cases, we found that the exceptions happened in some "extreme" situations, e.g., the distribution of ground-truth bears abnormal variation or the links that bear very big/small ground-truth values have super weak connectivity with other part of the topology, leaving the agent in *GSPC* no or very few options to take *Action-I* and *Action-U*. The average performance in different networks over all test runs is listed in the captions of Figs. 5.5 (a)$\sim$(d). From the results, we can conclude that on average *GSPC* significantly outperforms *Random* and *DP*.

GSPC demonstrates its superiority over *Random* and *DP* with few exceptions. Due to the involved complexity factors, e.g, the large size of topology, randomness

of ground-truth and monitor placement, we concluded that exceptions most likely happen in "extreme" situation such as the distribution of ground-truth bears abnormal variation or the entire group of links that bears very big/small ground-truth has super weak connectivity with other part of the topology, making *Action-I* and *Action-U* unable to develop great selection. Overall, the average performance of GSPC (average final TEB value by GSPC) on ground-truth-oriented stability and monitor-placement-oriented stability are shown in Table 5.3.

Table 5.3: Average GSPC Performance

Methods	w.r.t Ground-Truth	w.r.t Monitor Placement
Random	110351.393	107667.907
DP	61451.033	58505.287
GSPC	75900.273	42334.707

5.2 Evaluation of the GES algorithm

We compare the GES algorithm with the CTB algorithm. We test them over real-world ISP networks. In each network, we randomly select monitoring nodes and adopt the GSPC algorithm to construct measurement paths among those monitoring nodes. In each scenario, we simulate the behaviour of load balancers at randomly selected routers by setting the available bandwidth values among the links associated with a selected router to satisfy fairness constraints. Note that we require that a selected router should have at least two associated links because otherwise, the load balancer takes no effect. In the rest, we use $\mathcal{N}_{LB}$ to record the routers that pose fairness constraints. Regarding total capacity constraints, we sum up the maximum bandwidth values of all links associated with a router and set the total value as the constraint. Note that in practice, a link's maximum bandwidth value is available based on hardware specification.

We tested over four real-world topologies, including Abovenet (AS6461), EBONE (AS1755), Exodus (AS3967), and Tiscali (AS3257), whose topologies are from the Internet Topology Zoo (http://www.topology-zoo.org). The network parameters are listed in Table 5.4.

We use the total error bound (TEB), which is defined as the sum of error bounds of all links in the *min*-system, to compare the performance of different methods.

Table 5.4: Network Parameters

ISP name	$\|L\|$	$\|V\|$	$\|\mathcal{N}_{LB}\|$	R_{δ_v}	R_b	$\|\mathcal{M}\|$
Abovenet	294	182	82	$[0.90, 0.95]$	$[20, 300]$	82
EBONE	381	172	72	$[0.90, 0.95]$	$[20, 300]$	37
Exodus	434	201	100	$[0.90, 0.95]$	$[20, 300]$	39
Tiscali	404	240	120	$[0.90, 0.95]$	$[20, 300]$	90

δ_v denotes the threshold of fairness constraint of node v, R_{δ_v} the possible range where fairness threshold δ_v would be sampled, R_b the range of bandwidth values from which the ground-truth values are sampled, and $|\mathcal{M}|$ the number of monitors.

5.2.1 Performance Results

In all four real-world network topologies, finding out all the possible measurement paths between monitors is impractical (the problem of listing all MPs is NP-hard). With the GSPC algorithm, we would build effective MPs that help reduce the TEB. The number of measurement paths built with GSPC $|\mathcal{P}_{GSPC}|$ for different networks are displayed in Table 5.5.

For each network scenario with a given ϵ, we run GES 100 times, each with a random node update sequence, and record the smallest TEB as the final result. Note that the node update sequence and ϵ have no impact on CTB, so we only need to run CTB once for each scenario. The code is implemented in Python 3.11, and the experiments are executed on a laptop (6-Core Intel Core i7, 2.2 GHz, MEM: 16 GB, macOS Sonoma Version 14.4.1). Table 5.5 shows the performance comparison results between GES and CTB when ϵ is set to 0.1. The results demonstrate that with link correlation constraints, GES can significantly reduce the TEB, leading to over 70% TEB reduction over the CTB algorithm. In addition, GES can obtain more identifiable links than CTB.

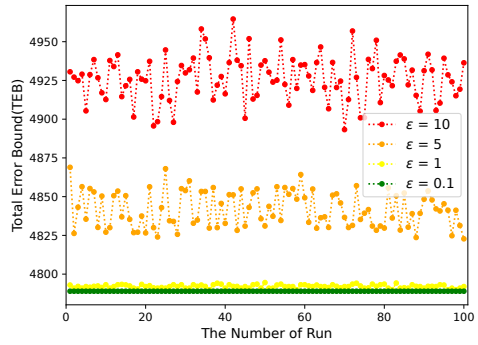
Table 5.5: Performance Comparison between GES and CTB

ISP name	$\lvert\mathcal{P}_{GSPC}\rvert$	$\lvert\mathcal{I}\rvert$ (GES/CTB)	TEB (GES/CTB)	TEB reduction(%)
Abovenet	5000	**123**/108	**4789.0**/16894	72%
EBONE	5000	**140**/127	**9079.4**/32139	72%
Exodus	6000	**124**/114	**12604.4**/49063	74%
Tiscali	6000	**155**/125	**11482.2**/47227	76%

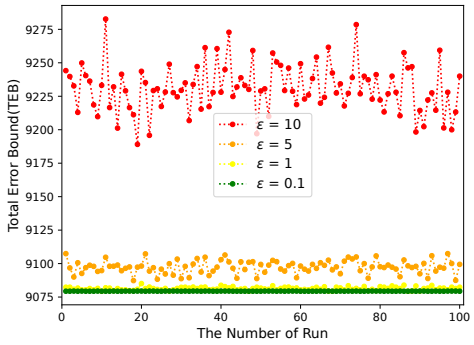
$|\mathcal{P}_{GSPC}|$ denotes the number of measurement paths built with GSPC. $|\mathcal{I}|$(GES/CTB) denote the number of identifiable links obtained with GES and CTB, respectively.

5.2.2 The Impact of ϵ

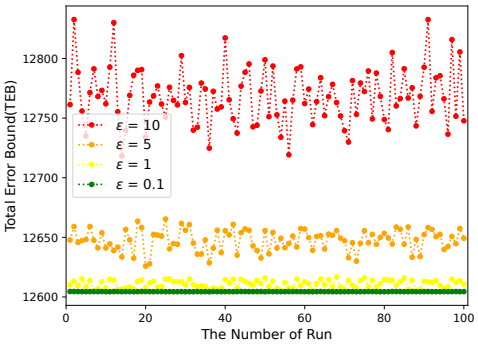
As we have mentioned before, there is interdependence among the links updates, and a minor adjustment to a link's bound may substantially influence the updates of other links in the *min*-system. We have used parameter ϵ to control this "ping-pong" effect. As shown in Figure 5.6, the fluctuation displayed from different node update sequences reveals that the potential links' improvements discarded by ϵ-LUAN at different nodes lead to different final results due to the "ping-pong" effects over



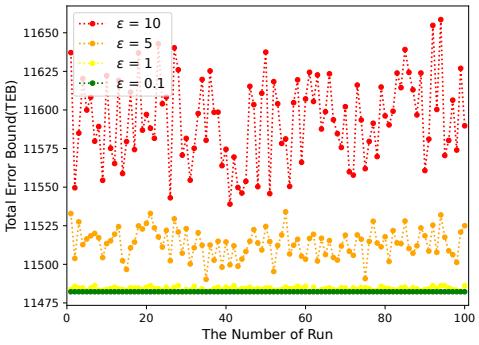
(a)Abovenet (US)



(b)EBONE (Europe)



(c) Exodus (US)



(d) Tiscali (Europe)

Figure 5.6: The performance of GES over different update sequences with different ϵ values.

the whole *min*-system. Nevertheless, with the decrease of ϵ, not only does the TEB reduce, but also the TEB fluctuates in a smaller range.

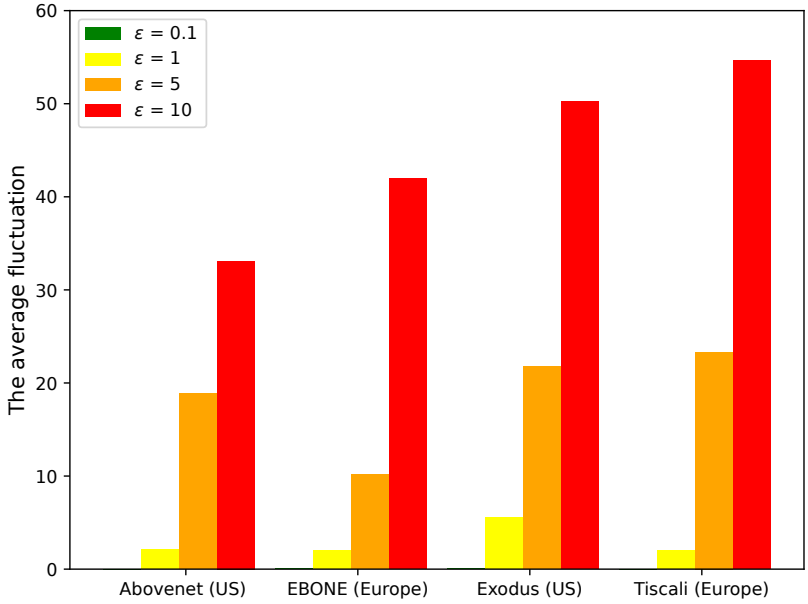


Figure 5.7: The average fluctuation under different ϵ values over 100 GES runs.

To further evaluate the impact of ϵ on the final results. We run GES multiple times, each run using a random node update sequence. We define a metric, average fluctuation, to account for the distance between the TEB of different runs and the minimum TEB obtained so far. That is, given ϵ and N runs of the GES algorithm, the average fluctuation is calculated as:

$$\frac{\sum_{i=1}^{N}(TEB_{(\epsilon,i)} - TEB_{est})}{N} \tag{5.1}$$

where $TEB_{(\epsilon,i)}$ is the TEB given ϵ at the ith run of GES and $TEB_{est} = \min_i TEB_{(\epsilon,i)}$. Figure 5.7 shows that the average fluctuation decreases as the decreasing of the ϵ value. When $\epsilon = 0.1$, the final results remain nearly unchanged.

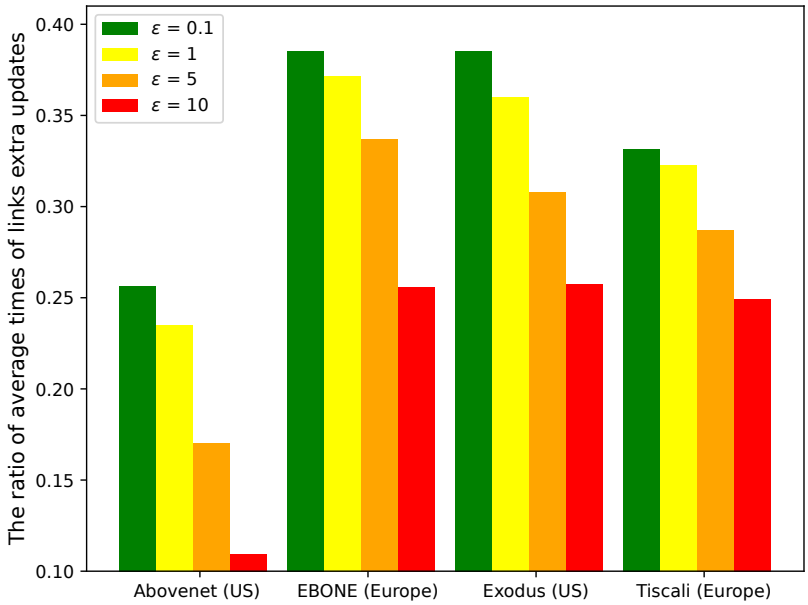


Figure 5.8: The ratio of the average times of links' extra updates under different ϵ values over 100 GES runs.

Figure 5.8 shows the ratio of the average times of links' extra updates, which is calculated as:

$$\frac{\sum_{i=1}^{N}(T_{(\epsilon,i)} - |\mathcal{L}_{unident}^{CTB}|)}{N|\mathcal{L}_{unident}^{CTB}|} \tag{5.2}$$

where $T_{(\epsilon,i)}$ is the total times of links updated given ϵ at the ith run of GES and $\mathcal{L}_{unident}^{CTB}$ is the set of unidentifiable links only derived from the CTB results of the corresponding *min*-system. Clearly, a smaller updating threshold ϵ leads to more frequent extra link updates. Figures 5.7 and 5.8 together indicate the tradeoff between the quality of performance results and the computation overhead.

5.3 Summary of Evaluation Results

As a quick summary of evaluation results over real-world ISP topology as well as simulated networks, we conclude that:

- *GSPC* improves two baseline path construction methods, *Random* and Diversity Preferred (*DP*), in terms of average error bound by 238% and 193%, respectively. *GSPC* also returns near-optimal results in small-scale simulated networks where listing all measurement paths for deriving the ground-truth global optimum is possible.

- GES can effectively utilize the new constraints to improve the performance bounds of CTB by over 70% in various simulated network scenarios.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This dissertation has explored the critical challenges and has presented innovative solutions in bandwidth tomography. The research presented in this dissertation provides a structured and algorithmic approach to identifying links's available bandwidth and deriving the tightest error bounds using measurement paths.

The primary contributions of this dissertation include the development of an efficient polynomial-time algorithm for identifying identifiable links and refining the lower and upper bounds of unidentifiable links. Through the introduction of the Global ϵ-Stabilizing (GES) algorithm, we have demonstrated how additional network constraints, such as fairness and total capacity constraints, can be leveraged to narrow down bandwidth estimates, thereby improving the accuracy and reliability of bandwidth tomography. Moreover, the Guided Sequential Path Construction (GSPC) method integrates reinforcement learning techniques with domain-specific knowledge to optimize path selection. It significantly reduces the computational complexity and improves accuracy in estimating link bandwidths. Focusing on inference methodologies, reinforcement learning-based path construction, and performance optimization techniques, this dissertation is the first attempt to systematically study the challenging inverse problem of a network *min*-system. These contributions, together, consist of an important chapter in bandwidth tomography research.

Our extensive evaluation, conducted over real-world ISP topologies and simulated networks, has confirmed the efficacy of the proposed solutions. The experimental results demonstrated that GSPC outperforms traditional path construction methods

such as Random and Diversity Preferred (DP) by improving average error bounds by 238% and 193%, respectively. Furthermore, our comparative analysis between GES and the Calculate the Tightest Bounds (CTB) benchmark demonstrates the advantage of incorporating additional constraints in refining performance bounds, yielding a 70% improvement over existing models.

One of the most fundamental findings of this dissertation is the computational complexity of finding the global tightest error bounds. The research proves that deriving the optimal solution requires an exhaustive enumeration of all possible measurement paths, a problem classified as #P-complete. This finding calls for a heuristic and learning-based approach, such as GSPC, to achieve near-optimal performance while maintaining computational feasibility.

The implications of this dissertation extend beyond bandwidth tomography. The methodologies developed in this dissertation have potential applications in broader domains, including network security, performance monitoring, and even financial transaction analysis in blockchain networks. The adaptability of the proposed techniques allows for integration with emerging network paradigms, such as In-band Network Telemetry (INT), to enhance real-time monitoring and diagnostics.

In conclusion, this dissertation provides a comprehensive framework for improving bandwidth tomography through novel algorithmic and learning-based techniques. The proposed models not only enhance bandwidth estimation but also lay the groundwork for future innovations in intelligent network monitoring and optimization. As the demand for efficient network diagnostics grows, the insights and methodologies developed in this dissertation will serve as a crucial foundation for advancing the field of network inference and performance estimation.

6.2 Future Work

Despite the significant advancements achieved in this research, several avenues remain open for future exploration, including:

1. **New Measurement Path Construction Methods:** One potential direction is the enhancement of GSPC with (1) deep reinforcement learning techniques and/or (2) new findings in the relationship between bottleneck and topological properties to further optimize path selection strategies dynamically. In addition, how to construct measurement paths for obtaining the “tightest" error bounds

under the constrained model assumed in Chapter 3 is still an open problem.

2. **Bandwidth tomography when the routing Matrix is unknown:** Current bandwidth tomography methods rely on the availability of a routing matrix, which describes the mapping between paths and individual network links. However, in real-world scenarios, this matrix is often unavailable due to security concerns, dynamic routing policies, or lack of administrative access. Future research should explore inference techniques that estimate bandwidth without relying on a predefined routing matrix. Possible approaches include blind signal separation (BSS), compressed sensing, and leveraging deep learning models to infer path-link relationships dynamically.

3. **Bandwidth tomography when the underlying network topology is partially known:** In many real-world networks, complete knowledge of the underlying topology is not always available, particularly in decentralized or large-scale environments such as peer-to-peer, overlay networks, and cloud infrastructures. Future research should investigate how to infer bandwidth performance when only partial topology information is available. Methods such as probabilistic graphical models, Bayesian inference, and reinforcement learning-based exploration strategies could be used to iteratively refine topology estimates while inferring bandwidth information with limited path knowledge.

4. **Bandwidth tomography when measurement results vary over time:** Most existing tomography methods assume static network conditions, which is often unrealistic in modern, dynamic networks where bandwidth fluctuates due to congestion, routing changes, and varying traffic loads. Future research should focus on designing adaptive bandwidth tomography methods that account for temporal variations. Approaches such as online learning, recurrent neural networks (RNNs), and Kalman filtering can be used to model time-dependent variations in network measurements and estimate the probabilistic properties of link-level bandwidth.

Bibliography

[1] Y. Vardi, "Network tomography: estimating source-destination traffic intensities from link data," *Journal of the American Statistical Association*, vol. 91, no. 433, pp. 365–377, 1996.

[2] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Inferring link metrics from end-to-end path measurements: identifiability and monitor placement," *IEEE/ACM Transactions on Networking (TON)*, vol. 22, no. 4, pp. 1351–1368, 2014.

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

[4] 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.

[5] E. Lawrence, G. Michailidis, V. Nair, and B. Xi, "Network tomography: a review and recent developments," *Ann Arbor*, vol. 1001, no. 48, pp. 109–1107, 2006.

[6] Y. Xia and D. Tse, "Inference of link delay in communication networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 2235–2248, 2006.

[7] Y. Bejerano and R. Rastogi, "Robust monitoring of link delays and faults in ip networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 5, pp. 1092–1103, 2006.

[8] R. Kumar and J. Kaur, "Practical beacon placement for link monitoring using network tomography," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 2196–2209, 2006.

[9] A. Gopalan and S. Ramasubramanian, "On identifying additive link metrics using linearly independent cycles and paths," *IEEE/ACM Transactions on Networking*, vol. 20, no. 3, pp. 906–916, 2012.

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

[11] 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*, 2013, pp. 581–590.

[12] 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*, vol. 24, no. 3, pp. 1392–1403, 2016.

[13] 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*, 2014, pp. 1447–1455.

[14] T. He, L. Ma, A. Swami, and D. Towsley, *Network tomography: identifiability, measurement design, and network state inference*. Cambridge University Press, 2021.

[15] N. G. Duffield, J. Horowitz, and F. L. Prestis, "Adaptive multicast topology inference," in *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No. 01CH37213)*, vol. 3. IEEE, 2001, pp. 1636–1645.

[16] N. G. Duffield, J. Horowitz, F. L. Presti, and D. Towsley, "Multicast topology inference from measured end-to-end loss," *IEEE Transactions on Information Theory*, vol. 48, no. 1, pp. 26–45, 2002.

[17] Y. Lin, T. He, S. Wang, K. Chan, and S. Pasteris, "Multicast-based weight inference in general network topologies," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[18] 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*, vol. 18, no. 1, pp. 123–135, 2009.

[19] Y. Lin, T. He, S. Wang, K. Chan, and S. Pasteris, "Looking glass of nfv: Inferring the structure and state of nfv network from external observations," *IEEE/ACM Transactions on Networking*, vol. 28, no. 4, pp. 1477–1490, 2020.

[20] C. Tebaldi and M. West, "Bayesian inference on network traffic using link count data," *Journal of the American Statistical Association*, vol. 93, no. 442, pp. 557–573, 1998.

[21] L. Ma, Z. Zhang, and M. Srivatsa, "Neural network tomography," *arXiv preprint arXiv:2001.02942*, 2020.

[22] Y. Qiao, K. Wu, and X. Yuan, "Autotomo: Learning-based traffic estimator incorporating network tomography," *IEEE/ACM Transactions on Networking*, vol. 32, no. 6, pp. 4644–4659, 2024.

[23] Y. Qiao, X. Yuan, and K. Wu, "Routing-oblivious network tomography with flow-based generative model," in *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications*, 2024, pp. 2139–2148.

[24] Y. Hu and L. Zhao, "DeepNT: Path-centric graph neural networks for network tomography," 2025. [Online]. Available: https://openreview.net/forum?id=pQOHbTpAwf

[25] C. Feng, J. An, K. Wu, and J. Wang, "Bound inference and reinforcement learning-based path construction in bandwidth tomography," *IEEE/ACM Transactions on Networking*, pp. 1–14, 2021.

[26] S. S. Chaudhari and R. C. Biradar, "Survey of bandwidth estimation techniques in communication networks," *wireless personal communications*, vol. 83, no. 2, pp. 1425–1476, 2015.

[27] K. Lai and M. Baker, "Measuring link bandwidths using a deterministic model of packet delay," in *Proceedings of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, 2000, pp. 283–294.

[28] N. Hu, L. Li, Z. M. Mao, P. Steenkiste, and J. Wang, "Locating internet bottlenecks: Algorithms, measurements, and implications," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 41–54, 2004.

[29] N. Bartolini, T. He, V. Arrigoni, A. Massini, F. Trombetti, and H. Khamfroush, "On fundamental bounds on failure identifiability by boolean network tomography," *IEEE/ACM Transactions on Networking*, vol. 28, no. 2, pp. 588–601, 2020.

[30] A. Rizk, "Network tomography using min-plus system theory," *Master's Thesis, TU Darmstadt*, 2008.

[31] A. Rizk and M. Fidler, "On the identifiability of link service curves from end-host measurements," in *International Conference on Network Control and Optimization*. Springer, 2008, pp. 53–61.

[32] F. Baccelli, G. Cohen, G. Olsder, and J. Quadrat, "Synchronization and linearity - an algebra for discrete event systems," *The Journal of the Operational Research Society*, vol. 45, 01 1994.

[33] J.-Y. Le Boudec and P. Thiran, *Network calculus: a theory of deterministic queuing systems for the internet*. Springer Science & Business Media, 2001, vol. 2050.

[34] L. Tan, W. Su, W. Zhang, J. Lv, Z. Zhang, J. Miao, X. Liu, and N. Li, "In-band network telemetry: a survey," *Computer Networks*, vol. 186, p. 107763, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128620313396

[35] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," 2016.

[36] Y. Qiao, K. Wu, and M. Khabbazian, "Non-intrusive and high-efficient balance tomography in the lightning network," in *ACM ASIACCS*, Hong Kong, June 2021.

[37] 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*. New York, NY, USA: ACM, 2013, pp. 391–404.

[38] L. G. Valiant, "The complexity of enumeration and reliability problems," *SIAM Journal on Computing*, vol. 8, no. 3, pp. 410–421, 1979.

[39] M. Jain and C. Dovrolis, "Pathload: a measurement tool for end-to-end available bandwidth," in *In Proceedings of Passive and Active Measurements (PAM) Workshop*. Citeseer, 2002.

[40] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 27–51, 2014.

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

[42] J. Xie and X. Wang, "A survey of mobility management in hybrid wireless mesh networks," *IEEE network*, vol. 22, no. 6, pp. 34–40, 2008.

[43] Y. Liu, K.-F. Tong, X. Qiu, Y. Liu, and X. Ding, "Wireless mesh networks in iot networks," in *2017 International workshop on electromagnetics: applications and student innovation competition*. IEEE, 2017, pp. 183–185.

[44] J. Burchard, D. Chemodanov, J. Gillis, and P. Calyam, "Wireless mesh networking protocol for sustained throughput in edge computing," in *2017 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2017, pp. 958–962.

[45] D. Zhang, M. Piao, T. Zhang, C. Chen, and H. Zhu, "New algorithm of multi-strategy channel allocation for edge computing," *AEU-International Journal of Electronics and Communications*, vol. 126, p. 153372, 2020.

[46] A. A. Pirzada and M. Portmann, "High performance aodv routing protocol for hybrid wireless mesh networks," in *2007 Fourth Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services (MobiQuitous)*. IEEE, 2007, pp. 1–5.

[47] T. Yang, Y. Jin, Y. Chen, and Y. Jin, "Rt-wabest: A novel end-to-end bandwidth estimation tool in ieee 802.11 wireless network," *International Journal of Distributed Sensor Networks*, vol. 13, no. 2, p. 1550147717694889, 2017.

[48] F. Boccardi, J. Andrews, H. Elshaer, M. Dohler, S. Parkvall, P. Popovski, and S. Singh, "Why to decouple the uplink and downlink in cellular networks and how to do it," *IEEE Communications Magazine*, vol. 54, no. 3, pp. 110–117, 2016.

[49] R. Jain, A. Durresi, and G. Babic, "Throughput fairness index: An explanation," in *ATM Forum contribution*, vol. 99, no. 45, 1999.

[50] A. B. Sediq, R. H. Gohary, R. Schoenen, and H. Yanikomeroglu, "Optimal tradeoff between sum-rate efficiency and jain's fairness index in resource allocation," *IEEE Transactions on Wireless Communications*, vol. 12, no. 7, pp. 3496–3509, 2013.

[51] F. Zabini, A. Bazzi, B. M. Masini, and R. Verdone, "Optimal performance versus fairness tradeoff for resource allocation in wireless systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 4, pp. 2587–2600, 2017.

[52] E. H. Ong and J. Y. Khan, "On optimal network selection in a dynamic multi-rat environment," *IEEE Communications Letters*, vol. 14, no. 3, pp. 217–219, 2010.

[53] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[54] B. Eva, "Principles of indifference," *The Journal of Philosophy*, vol. 116, no. 7, pp. 390–411, 2019.

[55] N. Spring, R. Mahajan, D. Wetherall, and H. Hagerstrom, "Rocketfuel: an isp topology mapping engine," 2002. [Online]. Available: http://research.cs.washington.edu/networking/rocketfuel/