2017

Inference in Networking Systems with Designed Measurements

Chang Liu

INFERENCE IN NETWORKING SYSTEMS WITH DESIGNED MEASUREMENTS

A Dissertation Presented

by

CHANG LIU

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

February 2017

College of Information and Computer Sciences

INFERENCE IN NETWORKING SYSTEMS WITH DESIGNED MEASUREMENTS

A Dissertation Presented

by

CHANG LIU

Approved as to style and content by:

Don Towsley, Chair

Ramesh Sitaraman, Member

Krista Gile, Member

Dan Sheldon, Member

Ting He, Member

James Allan, Chair
College of Information and Computer Sciences

ACKNOWLEDGMENTS

I thank my parents for their support. I thank my advisor Don, for his guidance and support through out the six and half years. I thank my committee members, for their help on this work.

I thank my lab mates who I worked and talked with everyday during my Ph.D. program. I want to thank Fabricio, Steve Chengxian Li, and Stephen Li, for helping me in many ways. My thanks goes to Mostafa for all the foosball games and the Championship we won.

I also thank my friends at Amherst and those scattering around the world, Haibin Huang, Steve Chengxian Li, Yahan Zhou and Ying Cao, Tian Guo, Pengyu Zhang, Wenzhe Wu, Weize Kong, Xiang Zhao, Hang Su, Zhaoliang Lun, Jingyi Guo, Xiaojian Wu, Jieqi Kang and Shan Lu, Ying Wang, Bingjie Wang, Qing Wei, Yuting Chen and Jin Zhao, Rui Chen, Meizhen Shi, Shengxin Zha, Jiangmeng Zhang, and many more that I cannot list all their names here.

To all of you,

Amherst, Amherst,
six and half years' the swiftest,
all my friends and dearest,
I will miss you from my deepest.

ABSTRACT

INFERENCE IN NETWORKING SYSTEMS WITH DESIGNED MEASUREMENTS

FEBRUARY 2017

CHANG LIU

B.S., XI'AN JIAOTONG UNIVERSITY

M.S., UNIVERSITY OF MASSACHUSETTS AMHERST

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Don Towsley

Networking systems consist of network infrastructures and the end-hosts have been essential in supporting our daily communication, delivering huge amount of content and large number of services, and providing large scale distributed computing. To monitor and optimize the performance of such networking systems, or to provide flexible functionalities for the applications running on top of them, it is important to know the internal metrics of the networking systems such as link loss rates or path delays. The internal metrics are often not directly available due to the scale and complexity of the networking systems. This motivates the techniques of inference on internal metrics through available measurements.

In this thesis, I investigate inference methods on networking systems from multiple aspects. In the context of mapping users to servers in content delivery networks,

we show that letting user select a server that provides good performance from a set of servers that are randomly allocated to the user can lead to optimal server allocation, of which a key element is to infer the work load on the servers using the performance feedback. For network tomography, where the objective is to estimate link metrics (loss rate, delay, etc.) using end-to-end measurements, we show that the information of each end-to-end measurement can be quantified by Fisher Information and the estimation error of link metrics can be efficiently reduced if the allocation of measurements on paths is designed to maximize the overall information. Last but not least, in the context of finding the most reliable path for routing from a source to a destination in a network while minimizing the cost of exploring lossy paths, the trade-off between exploiting the best paths based on estimated loss rates and taking the risk to explore worse paths to improve the estimation is investigated, and online learning methods are developed and analyzed. The performance of the developed techniques are evaluated with simulations.

TABLE OF CONTENTS

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

In large scale communication systems, such as the Internet, the cloud of Amazon EC2, or the Content Delivery Network of Akamai, internal metrics of the system, such as server workloads or link loss rates in the underlying network, are often desired yet not available due to the size and complexity of the networking system. For example, a central controller of a data center will want to know the load of each server for task scheduling so as to achieve load balancing, or the service provider of an Autonomous System network will want to know the failure rates of routers in the network for maintenance purposes. In these cases, inferences techniques are needed to estimate the internal metrics of interest.

There are several dimensions to this problem. One regards the study of inferences in networking systems as focusing on the development of estimators of the internal metrics for various applications. Another, however, focuses on design of the inference process, e.g., on how to allocate measurement budget on data collection, to improve efficiency of the inference.

This thesis studies the problem of inferences in networking systems with a focus on three different applications, and explores both the development of estimators and improvement of inference accuracy. The goal of this thesis is to advance our understanding of how inference techniques can be used as an aid for performance optimization in large scale networks, and how it can be done efficiently.

Inference techniques are needed in many application scenarios in large scale networks and systems. In data center management, for example, information such as

server utilization is required if one wants to achieve load balancing in the system. It's possible for a centralized controller to query all the servers and gather information about their utilizations, but such operations incur large amounts of communication overhead when the system is large. Furthermore, there are cases that the agent in need of such information does not have the access or authorization to directly ask servers for their status: imagine a client application that shares servers in a data center with other clients trying to guess the server loads so that to optimize its job scheduling. In some other cases, the information of interest cannot be obtained by querying components in the network but can only be estimated though measurements, examples of which include estimating link delays or server failure probabilities. These estimations also become complicated when network size grows large. When there are many links or servers in the network it becomes impossible to measure each link or server directly, and one may have to design estimation methods to obtain such information using more accessible end-to-end measurements. This thesis is motivated by these problems, and will investigate inference techniques in three different application scenarios.

The first part of the thesis investigates how to allocate servers to users in a content delivery network so as to achieve load balancing, where a key ingredient is to infer server loads through outside performance measurements. Content delivery networks (CDN) deliver much of the world's web and video content by deploying a large distributed network of servers. Server-to-user mapping is a key component that affects user experience for CNDs. On a global level, a CDN usually routes each user request to a cluster of servers located geographically close to the user. At a local level, within a cluster, the problem becomes that of how to map users to servers so as to achieve load balancing. We investigate this problem from the perspective of user-side server selection with the aid of server load inferences on the user side.

In the second part of the thesis, we study how to efficiently infer link loss probabilities in a network through end-to-end path loss measurements. Network tomography is the process of inferring the individual performance of networked elements (e.g., links) using network measurements conducted from the edge. Previous works have investigated how to place monitors in a network (enabling measurement functionality on nodes in the network) and what kind of measurement paths should be constructed between these monitors so that all the internal link metrics can be identified. But the problem remains as to how to allocate a limited measurement budget on end-to-end measurement paths to efficiently estimate the link metrics, a.k.a. measurement design. In this part of the thesis, we study the allocation of measurements among paths to minimize the estimation error given a measurement budget.

The third part of the thesis continues to look at network tomography on link loss probabilities, but with a focus on comparing difference measurement approaches. Two kind of end-to-end measurements can be used for network tomography: multicast measurement, and unicast measurement. We compare the two methods on tree topologies. Intuitively, multicast measurement that start from the root and traverses to all the leaf nodes in the tree has the advantage of covering all the links, yet unicast measurement has the flexibility of allocating measurements non-uniformly as needed across different part of the tree. This part of the thesis focuses on comparing the efficiency of these two measurement methods with analysis and experiments.

The fourth part of the thesis focuses on online routing with regard to finding the most reliable path in a network. In contrast to network tomography, the goal is not to determine the performance of all internal links, but only on identifying the best path from a set of paths in the network. We define path reliability as the probability a packet is delivered successfully across the path. We assume that the success across each link is a Bernoulli process independent from link to link, and that a path success probability is then the product of link success probabilities of links

on the path. The goal is to design an algorithm that allocates measurements across paths so that to find the path with the largest success probability while minimizing the cost of measurement, where cost is defined as the expected number of losses occuring during the measurement process. This part of the thesis studies the design of the measurement process for this problem and analyzes its performance.

1.1 Thesis Contributions

The following are the contributions of this thesis, summarized for each part as mentioned above.

1.1.1 Server Selection and Load Inference

We model and analyze a simple paradigm for client-side server selection where each user independently measures the performance of a set of candidate servers and selects the one that performs the best. Based on the inferred information about whether each server is overbooked by users more than its serving capacity, we design a simple algorithm and analyze its performance under the assumption that each user is provided at least two candidate servers to choose from. Our algorithms achieves system-wise load balancing while requiring no direct information about server load nor any coordination between servers and users. We prove the convergence of our algorithm and give an upper bound on the convergence time. We run simulations to evaluate our algorithms and demonstrate how design parameters will affect the performance.

1.1.2 Measurement Design for Link Loss Tomography

A framework is proposed to design probing experiments with a focus on probe allocation, and applying it to two concrete problems: packet loss tomography and packet delay variation (PDV) tomography. Based on Fisher Information Matrix (FIM), this work designs the distribution of probes across paths to maximize the best accuracy

of unbiased estimators, asymptotically achievable by the maximum likelihood estimator. Two widely-adopted objective functions are considered: the determinant of the inverse FIM (D-optimality) and the trace of the inverse FIM (A-optimality). The A-optimal criterion is then extended to incorporate heterogeneity in link priorities. Under certain conditions on the FIM, satisfied by both loss and PDV tomography, we derive explicit expressions for both objective functions. When the number of probing paths equals the number of links, these lead to closed-form solutions for the optimal design; when there are more paths, we develop heuristics to select a subset of paths and optimally allocate probes within the subset. Observing the dependence of the optimal design on unknown parameters, we further propose an algorithm that iteratively updates the design based on parameter estimates, which converges to the design based on true parameters as the number of probes increases. Using packet-level simulations on real datasets, we verify that the proposed design effectively reduces estimation error compared with the common approach of uniformly distributing probes.

1.1.3 Comparing Multicast Measurement and Unicast Measurement for Link Loss Tomography

We compare the performance of using unicast measurement with that of using multicast measurements for link loss tomography. We focus on the link loss tomography problem and networks with tree topologies for convenience of comparison. To theoretically compare multicast and unicast, an observation model for multicast and expressions for calculating the Fisher Information Matrix are developed. We apply optimal experiment design and derive a simplified solution of probe allocation for unicast measurement. Using a packet level simulator, we evaluate and compare the per-link MSE of multicast and unicast under varying parameter settings including link weights, link success rates and tree size. Our results show that multicast measurements often outperforms unicast measurement, though unicast can outperform

multicast under a tight probing budget constraint, especially when one is interested in minimizing a weighted average of per-link MSEs. Furthermore, multicast achieves more consistent performance with respect to varying link success rates or tree size.

1.1.4 Online Routing with Inferred Path Reliability

The problem of finding the most reliable path in a network is modeled as a Multi-Armed Bandit (MAB) problem where each arm represents a path, and the objective is to optimize the quality of communication between a source and a destination through adaptive path selection. We investigate two different measurement models. In the first model, link states are not directly observable. Sending a probe over a path results in either a success or a loss. When the latter occurs, no information is revealed as to which link might have dropped the probe. In the second model, a loss reveals the location of the link where the loss occurred. The objective is to find the most path with the largest success probability while minimizing the cost introduced by measuring sub-optimal paths. We design algorithms determining measurement allocation strategies on the set of paths, and develop performance bounds for the algorithms. Our results bring insights on the benefit of having additional information about the location of losses.

1.2 Thesis Outline

The rest of the thesis is organized as follows. In Chapter 2, I provide the context to this thesis work with background on network modeling, statistical inferences, and Multi-Armed Bandit problems. Chapter 3 formulates the problem of mapping server to users in the content delivery network, and proposes a method to infer congestion events at the server side, based a which an randomized algorithm is developed for users to select servers. Chapter 4 considers the problem of loss tomography where link loss rates are inferred from end-to-end path loss measurements, and demon-

strate the benefit of allocating measurement budget across paths based on the Fisher information provided by each path. Chapter 5 compares the performance of link loss tomography using end-to-end path measurements against that using multicast measurements. Simulation results show how the preference of the two measurement methods should depend on varies settings of the network parameters. Chapter 6 describes the algorithms for online routing with regard to find the most reliable path and present analysis and evaluation of the algorithms. We conclude in Chapter 7 and discuss future directions.

CHAPTER 2

DEFINITIONS AND BACKGROUND

In this chapter we define terms and notation commonly used in this dissertation. In addition, we provide an overview of statistical inference and Maximum Likelihood Estimation (MLE), which is relevant to all of the following chapters. We also introduce Fisher Information (a measure to quantify the statistical information in a sample) and explain how it relates to lower bounds on estimation errors through the Cramér-Rao bound. This definition and relationship are relevant to Chapter 4 and Chapter 5. Last, we provide an overview of Multi-Armed Bandit problems and algorithms, which is relevant to Chapter 6.

2.1 Definitions

Let $G = (V, E)$ be the undirected graph representing the network topology, where V is a set of vertices (or nodes) and E is a set of unordered pairs of vertices $l = (u, v), u \in V, u \in V$ representing a connection from u to v (a.k.a. links).

2.2 Statistical Inference and MLE

Statistical inference is the process of deducing properties of an underlying distribution by analysis of data. Statistical inference makes propositions about a population, using data drawn from the population using some form of sampling. Given a hypothesis about a population, for which we wish to draw inferences, statistical inference consists of (firstly) selecting a statistical model of the process that generates the data and (secondly) deducing propositions from the model. In statistics, an estimator is a

rule for calculating an estimate of a given quantity based on observed data. In this thesis, we only use point estimators, which is the rule to calculate a particular value that best approximates some parameter of interest. Maximum likelihood estimation is a type of estimation that maximize the likelihood of making the observations given the parameters.

Suppose there is a sample $x_1, x_2, \ldots, x_n$ of n independent and identically distributed observations, coming from a distribution that we assume to have a probability density function $f(\cdot|\theta)$, where θ is a vector of parameters in the parametric model of the distribution. The true value of θ is unknown and thus it is desirable to find an estimator $\hat{\theta}$ which could be as close to the true value as possible. The likelihood function of parameter θ given the set of samples is

$$\mathcal{L}(\theta; x_1, x_2, \ldots, x_n) = f(x_1, x_2, \ldots, x_n|\theta). \tag{2.1}$$

The Maximum Likelihood Estimator (MLE) is defined as,

$$\hat{\theta}_{MLE} = \arg\max_{\theta \in \Theta} \mathcal{L}(\theta; x_1, x_2, \ldots, x_n). \tag{2.2}$$

Although MLEs have no optimum properties for finite samples, it possesses a number of attractive limiting properties. One of them is the asymptotic efficiency (under regularity conditions [49]), i.e. it achieves the Cramér-Rao Bound when the number of sample size increases to infinity. We will introduce the Cramér-Rao Bound in the next section.

2.3 Fisher Information and Cramér-Rao Bound

Fisher information is one way to measure the statistical information contained in an observable random variable X about an unknown parameter θ that models the distribution of X. Let $f(X; \theta)$ be the likelihood function for θ, i.e., the probability

density of the random variable X conditional on the value of θ. The Fisher information $I(\theta)$ is defined as the variance of the score function, where the score function is defined as the gradient of the natural logarithm of $f(X;\theta)$ w.r.t. parameter θ. Under weak regularity conditions [12], the expected value of the score function is zero. Hence, the Fisher Information is the second moment of the score, expressed as

$$I(\theta) = \mathbb{E}\left[(\frac{\partial}{\partial\theta}\log f(X;\theta))^2|\theta\right], \tag{2.3}$$

where the expectation is taken with respect to the distribution of X given θ, i.e., $f(X;\theta)$. When $\theta = (\theta_1, \theta_2, \ldots, \theta_W)$ is a vector, the Fisher Information takes the form of a square matrix of dimension W where its elements are defined as

$$I_{i,j}(\theta) = \mathbb{E}\left[(\frac{\partial}{\partial\theta_i}\log f(X;\theta))(\frac{\partial}{\partial\theta_j}\log f(X;\theta))|\theta\right]. \tag{2.4}$$

The Cramér-Rao Bound relates the Fisher Information to the estimation error of any unbiased estimator $T(X) = (T_1(X), T_2(X), \ldots, T_W(X))$:

$$\mathrm{cov}_\theta(T_i(X), T_j(X)) \succeq I^{-1}(\theta), \tag{2.5}$$

where $\succeq$ means that $\mathrm{cov}(T(X)) - I^{-1}(\theta)$ is a positive semi-definite matrix. Furthermore,

$$(\mathrm{cov}_\theta(T(X)))_{i,i} = Var(T_i(X)) \geq I^{-1}_{i,i}(\theta). \tag{2.6}$$

To achieve the Cramér-Rao Bound the estimator must be efficient, which makes the Maximum Likelihood Estimator a great candidate because of its asymptotic efficiency property.

2.4 Multi-Armed Bandit problems and algorithms

In a Multi-Armed Bandit (MAB) problem, a forecaster is given a number of arms (or actions) K and a number of rounds T. For each round t, nature generates a reward

vector $r_t = (r_{1,t}, \ldots, r_{K,t}) \in [0,1]^K$ unobservable to the forecaster[1]. The forecaster chooses an arm $I_t \in 1, \ldots, K$ and receives payoff $r_{I_t,t}$, with the other rewards hidden. A general performance metric for solutions to the MAB problems is the regret,

$$R(T) = \mathbb{E}\left[\sum_{t=1}^{T} r_{*,t}\right] - \mathbb{E}\left[\sum_{t=1}^{T} r_{I_t,t}\right], \tag{2.7}$$

where $*$ represents the index of the best arm. Thus, the regret is the gap between the expected reward when using the best arm at each time step and the expected reward achieved by the solution. The goal is to maximize the cumulative rewards obtained, which is equivalent to minimizing the regret.

MAB problems can be classified according to how the reward vector is generated. In stochastic bandit problems, each entry $r_{i,t}$ in the reward vector is sampled independently, from an unknown distribution v_i, regardless of t. In adversarial bandit problems, the reward vector r_t is chosen by an adversary which, at time t, knows the past, but not I_t. In this thesis, however, we focus on non-adversarial stochastic MAB problems, where the underlying unknown distributions $v_i, i = 1, 2, \ldots, K$ do not change over time t.

[1]In general rewards can be normalized to interval $[0, 1]$

CHAPTER 3

SERVER SELECTION WITH INFERENCE OF OVERBOOKING

3.1 Introduction

Modern content delivery networks (CDNs) host and deliver a large fraction of the world's web content, video content, and application services on behalf of enterprises that include most major web portals, media outlets, social networks, application providers, and news channels [39]. CDNs deploy large numbers of servers around the world that can store content and deliver that content to users who request it. When a user requests a content item, say a web page or a video, the user is directed to one of the CDN's servers that can serve the desired content to the user. The goal of a CDN is to maximize the performance perceived by the user while efficiently managing its server resources.

A key function of a CDN is *server selection* by which client software running on the user's computer or device, such as media player or a browser, is directed to a suitable server of a CDN [16]. The desired outcome of server selection is that each user is directed to a server that will provide the requested content with good performance. The performance metrics that are optimized vary by the content type. For instance, good performance for a user accessing a web page might mean low latency web page downloads. Good performance for a user watching a video might mean high bitrate video delivery by the server while avoiding video freezing and rebuffering [29].

Server selection can be performed in two distinct approaches that are not mutually exclusive. The first approach relies on *network-side server selection* algorithms to

monitor the real-time characteristics of the CDN and the Internet. Such algorithms are often complex and measure liveness and CDN server load, as well as latency, loss, and bandwidth of the communication paths between servers and users. Using this information, the algorithm computes a good "mapping" of users to servers, such that each user is assigned a "proximal" server capable of serving that user's content [39]. This mapping is computed periodically and is typically made available to the client using the domain name system (DNS). Specifically, the user's browser or media player looks up the domain name of the content that it wants to download and receives as translation the IP address of the selected server.

A complementary approach to network-side server selection commonly used is *client-side server selection* where the client embodies a server selection algorithm. The client software is typically unaware of the global state of the server infrastructure, the Internet, or other clients. Rather, the client software typically makes future server selection decisions based on its own historical performance measurements from past server downloads. Client-side server selection can often be implemented as a plug-in within media players, web browsers, and web download managers [3].

While client-side server selection can be used to select servers within a single CDN, it can also be used in a multi-CDN setting. Large content providers often make the same content available to the user via multiple CDNs. In this case, the client can try out the different CDNs and choose the "best" server from across multiple CDNs. For instance, NetFlix uses three different CDNs and the media player incorporates a client-side server selection algorithm to choose the "best" server (and the corresponding CDN) using performance metrics such as achievable video bitrates [1]. Note also that in a typical multi-CDN case, both network-side and client-side server selection can be used together, where the former is used to choose the candidate servers from each CDN and the latter is used by the user to pick the "best" among all the candidates.

3.1.1 The Go-With-The-Winner paradigm

A common and intuitive paradigm that is often used for client-side server selection in practice is what we call *"Go-With-The-Winner"*. It consists of an initial *trial period* during which each user independently "tries out" a set of *candidate servers* by requesting content or services from them (cf. Figure 3.1). Subsequently, each user independently *decides* on the "best" performing server using historical performance information that the user collected for the candidate servers during the trial period. It is commonly implemented in the content delivery context that incorporate selecting a web or video content server from among a cluster of such servers.

Besides content delivery, the Go-With-The-Winner paradigm is also used for other Internet services, though we do not explicitly study such services in this thesis. For instance, BIND, which is the most widely deployed DNS resolver (i.e., DNS client) on the Internet, tracks performance as a smoothed value of historical round trip times (called SRTT) from past queries for a set of candidate name servers. BIND then chooses a particular name server to query in part based on the computed SRTT values [31]. It is also notable that BIND implementations incorporate randomness in the candidate selection process.

The three key characteristics of the Go-With-The-Winner paradigm are as follows.

1. *Distributed control.* Each user makes decisions in a distributed fashion using only knowledge available to it. There is no explicit information about the global state of the servers or other users, beyond what the user can infer from it's own historical experience.

2. *Performance feedback only.* There is no explicit feedback from a server to a user who requested service beyond what can be inferred by the performance experienced by the user.

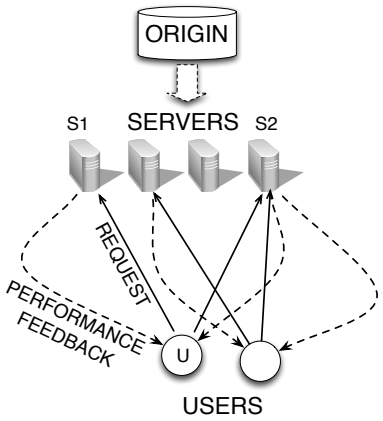


Figure 3.1: Client-side Server Selection with the Go-With-The-Winner paradigm. User U makes request to two candidate servers $S1$ and $S2$. After a trial period of observing the performance provided by the candidate, the user selects the better performing server.

3. *Choosing the "best" performer.* The selection criteria is based on historical performance measured by the user and consists of selecting the best server according to some performance metric (i.e., go with the winner).

Besides its inherent simplicity and naturalness, the paradigm is sometimes the only feasible and robust solution. For instance, in many settings, the client has no detailed knowledge of the state of the server infrastructure as it is managed and owned by other business entities. In this case, the primary feedback mechanism for the client is its own historical performance measurements.

While client-side server selection is widely implemented, its theoretical foundations are not well understood. A goal of our work is to provide such a foundation in the context of web and video content delivery. *It is not our intention to model a real-life client-side server selection process in its entirety which can involve other adhoc implemention-specific considerations. But rather we abstract an analytical model that we can explore to extract basic principles of the paradigm that are applicable in a broad context.*

3.1.2 Our contributions

We propose a simple theoretical model for the study of client-side server selection algorithms that use the Go-With-The-Winner paradigm. Using our model, we answer foundational questions such as how does randomness help in the trial period when selecting candidate servers? How many candidate servers should be selected in the trial phase? How long does it take for users to narrow down their choice and decide on a single server? Under what conditions does the selection algorithm converge to a state where all users have made correct server choices, i.e., selected servers provide good performance to their users? Some of our key results that help answer these questions follow.

(1) In Section 3.2, in the context of web content delivery, we analyze a simple algorithm called GoWithTheWinner where each user independently selects two or more random servers as candidates and decides on the server that provides the best cache hit rate. We show that with high probability, the algorithm converges quickly to a state where no cache is overloaded and all users obtain a 100% hit rate. Furthermore, we show that two or more random choices of candidate servers are necessary, as just one random choice will result in some users (and some servers) incurring cache hit rates that tend to zero, as the number of users and servers tend to infinity. This work is the first demonstration of the "power of two choices" phenomena in the context of client-side server selection for content delivery, akin to similar phenomena observed in balls-into-bins games [35], load balancing, circuit-switching algorithms [14], relay allocation for services like Skype [38], and multi-path communication [27].

(2) In Section 3.3, in the context of video content delivery, we propose a simple algorithm called MaxBitRate where each user independently selects two or more random servers as candidates and decides on the server that provides the best bitrate for the video stream, We show that with high probability, the algorithm converges quickly to a state where no server is overloaded and all users obtain the required

bitrate for their video to play without freezes. Further, we show that two or more random choices of candidate servers are necessary, as just one random choice will result in some users receiving bitrates that tend to zero, as the number of users and servers tends to infinity.

(3) In Section 3.4, we go beyond our theoretical model and simulate algorithm GoWithTheWinner in more complex settings. We establish an inverse relationship between the length of the history used for hitrate computation (denoted by τ) and the failure rate defined as the probability that the system converges to a non-optimal state. We show that as τ increases the convergence time increases, but the failure rate decreases. We also empirically evaluate the impact of the number of choices of candidate servers. We show that two or more random choices are required for all users to receive a 100% hitrate. Though even if only 70% of the users make two choices, it is sufficient for 95% of the users to receive a 100% hitrate. Finally, we show that the convergence time increases with system load. But, convergence time decreases when the exponent of power law distribution that describes content popularity increases.

3.2 Hit Rate Maximization for Web Content

The key measure of web performance is *download time* which is the time taken for a user to download a web object, such as an html page or an embedded image. CDNs enhance web performance by deploying a large number of servers in access networks "close" to the users. Each server has a cache capable of storing web objects. When a user requests an object, such as a web page, the user is directed to a server that can serve the object (cf. Figure 3.1). If the server already has the object in its cache, i.e, the user's request is a *cache hit*, the object is served from the cache to the user. In this case, the user experiences good performance, since the CDN's servers are proximal to the user and the object is downloaded quickly. However, if the requested object is not in the server's cache, i.e., the user's request is a *cache miss*, then the server

first fetches it from the origin, places it in its cache, and then serves the object to the user. In the case of a cache miss, the performance experienced by the user is often poor since the origin server is typically far away from the server and the user. In fact, if there is a cache miss, the user would have been better off not using the CDN at all, since downloading the content directly from the content provider's origin would likely have been faster! Since the size of a server's cache is bounded, cache misses are inevitable. A key goal of server selection for web content delivery is to jointly orchestrate server assignment and content placement in caches such that the cache hit rate is maximized. While server selection in CDNs is a complex process [39], we analytically model the key elements that relate to content placement and cache hit rates, leaving other factors that impact performance such as server-to-user latency for future work.

3.2.1 Problem Formulation

Let U be a set of n_u users who each requests an object picked independently from a set C of size n_c using a popularity distribution $\{p_1, p_2, \ldots, p_{n_c}\}$, where the k-th most popular object in C is picked with probability p_k. The user then makes a sequence of requests for that content item to the set of available servers. In practice, users tend to stay with one website for a while, say reading the news or looking at a friend's posts. We model the sequence of requests generated by each user as a Poisson process with homogeneous arrival rate λ. Note that each request from user u can be sent to one or more servers selected from $S_u \subseteq S$, where S_u is the set of candidate servers for user u.

Let S be the set of n_s servers that are capable of serving content to the users. Each server can cache at most κ objects and a cache replacement policy such as LRU is used to evict objects when the cache is full. Given that the download time of a web object is significantly different when the request is a cache hit versus a cache miss,

we make the assumption that the user can reliably infer if its request to download an object from a server resulted in a cache hit or a cache miss immediately after the download completes.

The objective of client-side server selection is for each user $u \in U$ to independently select a server $s \in S$ using only the performance feedback obtained on whether each request was a hit or a miss. Let the hit rate function $H(u, s, t)$ denote the probability of user u receiving a hit from server $s \in S_u$ at time t. We define the system-wide performance measure $H(t)$, as the best hit rate obtained by the worst user at time t,

$$H(t) \triangleq \min_{u \in U} \max_{s \in S_u} H(u, s, t), \tag{3.1}$$

a.k.a. the *minmax hit rate*. Our goal is to maximize $H(t)$. In the rest of the section, we describe a simple "Go-With-The-Winner" algorithm for server selection and show that it converges quickly to an optimal state, with high probability.

Note: Our formulation is intentionally simple so that it can model a variety of other situations in web content delivery. For instance, a single server could in fact model a cluster of front-end servers that share a single backend object cache. A single object can model a bucket of objects that cached together as is often done in a CDN context [39].

3.2.2 The *GoWithTheWinner* Algorithm

After each user $u \in U$ selects a content item and a set of σ servers S_u, the user executes algorithm *GoWithTheWinner* to select a server likely to always have the content. In this algorithm, each user locally executes a simple "Go-With-The-Winner" strategy of trying out σ randomly chosen candidate servers initially. For each server $s \in S_u$, the user keeps track of the most recent request results in a vector $\mathbf{h}^s = (h_1^s, h_2^s, \cdots, h_\tau^s)$ where $h_k^s = 1$ corresponds to the k-th recent request resulting in a hit from server s and $h_k^s = 0$ if otherwise. τ is the "sliding window size". Using

Algorithm 1: GoWithTheWinner

The current user u chooses a set of σ candidate servers $S_u \subseteq S$ uniformly at random from all the servers;
for *each* $s \in S_u$ **do**
    set $\mathbf{h^s} \leftarrow (h_1^s, h_2^s, \cdots, h_\tau^s) = \mathbf{0}$;
end
for *each arrival of request* **do**
    set t to the current time;
    Request content a_u from *all* servers $s \in S_u$;
    for *each server* $s \in S_u$ **do**
        $h_i^s \leftarrow h_{i-1}^s, 2 \leq i \leq \tau$;
        $h_1^s \leftarrow$ if hit; $h_1^s \leftarrow 0$, if miss;
        compute hit rate $H_\tau(u, s, t) \leftarrow (\sum_{i=1}^{\tau} h_i^s)/\tau$;
        if $H_\tau(u, s, t) = 100\%$ **then**
            decide on server s by setting $S_u \leftarrow \{s\}$;
            return;
        end
    end
end

the hit rates, each user then independently either chooses to continue with all the servers in S_u or decides on a single server that provided good performance. If there are multiple servers providing 100% hit rate, the user decides to use the first one found.

3.2.3 Analysis of Algorithm GoWithTheWinner

Here we analyze the case where $n_u = n_c = n_s = n$ and experimentally explore other variants where n_c and n_u are larger than n_s in Section 3.2.4 and 3.4. Let $H(t)$ be as defined in (3.1). If $\sigma \geq 2$, we show that with high probability $H(t) = 100\%$, for all $t \geq T$, where $T = O(\frac{\kappa}{\log(\kappa+1)}(\log n)^{\kappa+1} \log\log n)$. That is, the algorithm converges quickly with high probability to an optimal state where *every* user has decided on a single server that provides a 100% hit rate, and *every* server has the content requested by its users.

Definitions. A server s is said to be *overbooked* at some time t if users request more than κ distinct content items from server s, where κ is the number of content items a server can hold. Note that a server may have more than κ users and not be overbooked, provided the users collectively request a set of κ or fewer content items. Also, note that a server that is overbooked at time t is overbooked at every $t' \leq t$ since the number of users requesting a server can only remain the same or decrease with time. Finally, a user u is said to be *undecided* at time t if $|S_u| > 1$ and is said to be *decided* if it has settled on a single server to serve its content and $|S_u| = 1$. Note that each user starts out undecided at time zero, then decides on a server at some time t and remains decided in all future time later than t. Users calculate the hit rates of each of the available servers based on a history record of the last τ requests, where τ is called the sliding window size.

Lemma 1. *If the sliding window size $\tau = \Theta(\log^{\kappa+1} n)$, the probability that some user $u \in U$ decides on an overbooked server $s \in S_u$ upon any request arrival is at most $1/n^{\Omega(1)}$.*

Proof. If user u decides on server s then the current request together with the previous $\tau - 1$ requests are all hits. Let H_k, $k = 1, 2, \cdots, \tau$ be Bernoulli random variables, s.t. $H_k = 1$ if the most recent k-th request of u is a hit and $H_k = 0$ if it is a miss. To prove Lemma 1 we need to show

$$\mathbb{P}\left(\cap_{k=1}^{\tau}(H_k = 1)\right) \leq n^{-\Omega(1)}. \tag{3.2}$$

Let t_1 denote the time of the most recent request for content a_u from user u appears at server s, resulting in feedback H_1 to the user. Let $t_1 - \Delta$ be the time that the previous request for a_u arrives at s. Let $A_s = \{a_1, a_2, \cdots, a_M\}$ be the set of different content items requested at s, where $M > \kappa$. Let $N_i \geq 1$ be the number of users requesting a_i from s. WLOG, let $a_1 = a_u$ be the content that u requests, such that N_1 is the

number of users requesting for a_u. Because we assume all the users generates requests with a Poisson process with arrival rate λ, the aggregated arrival rate of requests for a_u is then $N_1\lambda$. Thus Δ is an exponential random variable, $\Delta \sim Exp(N_1\lambda)$. Now we look at the number of different requests arrives between time $t_1 - \Delta$ and t_1. Let $X_i, i = 2, 3, \cdots, M$ be an indicator that a request for a_i arrives at server s during the time interval $(t_1 - \Delta, t_1)$, we have $X_i \sim Bernoulli(1 - e^{-N_i\lambda\Delta})$. Furthermore, let random variable $Y = \sum_{i=2}^{M} X_i$ be the number of different requests arrived in the time interval. With the server running on LRU replacement policy,

$$\mathbb{P}(H_1 = 0) = \mathbb{P}(Y \geq \kappa), \tag{3.3}$$

because for content a_u to be swapped out of the server, more than κ different requests other than that for a_u must have arrived. Equation (3.3) shows that H_1 only depends on the number different requests arrived after the previous request for a_u, which means events $H_k, k = 1, 2, \cdots, \tau$ are mutually independent.

Furthermore[1], because $N_i \geq 1$, we have $X_i \geq_d X'$ where $X' \sim Bernoulli(1 - e^{-\lambda\Delta})$. Thus,

$$Y = \sum_{i=2}^{M} X_i \geq_d \sum_{i=2}^{M} X' = Z,$$

where $Z \sim Binomial(\kappa, (1 - e^{-\lambda\Delta}))$.

Thus, we have

[1]random variables $U \geq_d V$ if $\mathbb{P}(U > x) \geq \mathbb{P}(V > x)$ for all x.

$$\begin{aligned}\mathbb{P}(Y \geq \kappa) &\geq \mathbb{P}(Z \geq \kappa) \\ &= \int_0^\infty \mathbb{P}(Z \geq \kappa | \Delta = t) f_\Delta(t) dt \\ &= \int_0^\infty (1 - e^{-\lambda t})^\kappa N \lambda e^{-N\lambda t} dt \\ &= \frac{N!\kappa!}{(N+\kappa)!} \\ &\geq (N+\kappa)^{-\kappa},\end{aligned}$$

where $f_\Delta(t)$ is the probability density function of Δ.

Note that N is the number of users requesting a at server s, and is bounded by $N = O(\frac{\log n}{\log \log n})$, with high probability [42].

Now, we can finally prove (3.2). Let c' be an appropriate constant,

$$\begin{aligned}\mathbb{P}(\cap_{k=1}^\tau (H_k = 1)) = \mathbb{P}(H = 1)^\tau &= (1 - \mathbb{P}(H = 0))^\tau \\ &= (1 - \mathbb{P}(Y \geq \kappa))^\tau \\ &\leq (1 - (N+\kappa)^{-\kappa})^\tau \\ &\leq (1 - (c' \frac{\log n}{\log \log n} + \kappa)^{-\kappa})^\tau,\end{aligned}$$

which is $n^{-\Omega(1)}$ when $\tau = \Theta(\log^{\kappa+1} n)$. □

By bounding the time for τ requests to arrive at user u, we have the following,

Lemma 2. *If user u (with candidate servers S_u) is not decided at time t, then the server is overbooked at time $t - \delta$ for $\delta = \frac{\tau+1}{\lambda} c_0$ where $c_0 > 1$ is a constant, with high probability.*

Proof. Let random variable N_δ be the number of requests from u during time $(t - \delta, t)$, $N_\delta \sim Poisson(\lambda\delta)$. A bound on the tail probability of Poisson random variables is developed in [36] as

$$\mathbb{P}(X \leq x) \leq \frac{e^{-\lambda'}(e\lambda')^x}{x^x},$$

where $X \sim Poisson(\lambda')$ and $x < \lambda'$.

We can show there are at least $\tau + 1$ requests during $(t - \delta, t)$ w.h.p. as the following,

$$\begin{aligned}
\mathbb{P}\left(N_\delta < \tau + 1\right) &\leq e^{-\lambda\delta}\frac{(e\lambda\delta)^{\tau+1}}{(\tau+1)^{\tau+1}} = e^{-(\tau+1)c_0}(ec_0)^{(\tau+1)} \\
&= e^{-(\tau+1)(c_0-1)}c_0^{(\tau+1)} \\
&= n^{-\frac{(\tau+1)}{\log n}(c_0-1-\log c_0)} \\
&= n^{-\Theta(\log^\kappa n)},
\end{aligned}$$

as $c_0 > 1$ and $\tau = \Theta(\log^{\kappa+1} n)$. Thus, w.h.p. no fewer than $\tau + 1$ requests arrive at u. And because user is not decided at time t we know that with high probability, at least one of the previous τ requests results in a miss, which means that between the previous $(\tau + 1)$-th request and the miss, κ different other requests arrived at the server. Thus server s is *overbooked* at the time the previous $(\tau + 1)$-th request arrives, which with high probability is no earlier than $t - \delta$. □

Based on Lemmas 1 and 2, we can then establish the following theorem about the performance of Algorithm *GoWithTheWinner*.

Theorem 3. *With probability at least* $1 - \frac{1}{n^{\Omega(1)}}$*, the minmax hit rate* $H(t) = 100\%$ *for all* $t \geq T$*, provided* $\sigma \geq 2$ *and* $T = O(\frac{\kappa}{\log(\kappa+1)}(\log n)^{\kappa+1}\log\log n)$*. That is, with high probability, algorithm GoWithTheWinner converges by time* T *to an optimal state where each user* $u \in U$ *has decided on a server* $s \in S$ *that serves it content with a* 100% *hit rate.*

Proof. For simplicity, we prove the situation where $\sigma = 2$, i.e., each user initially chooses two random candidate servers in step 1 of the algorithm. The case where $\sigma > 2$ is analogous. Wlog, we also assume κ is at most $O(\log n/\log\log n)$, which includes the interesting case of κ equal to a constant. When the server capacity

is larger, i.e., if $\kappa = \Omega(\log n / \log\log n)$, there will be no overbooked servers with high probability and the theorem holds trivially. This observation follows from a well-known result that if n balls (i.e., users) uniformly and randomly select k out of n bins (i.e. servers), then the maximum number of users that select a server is $O(\log n / \log\log n)$ with high probability, when k is a fixed constant [42].

In contradiction to the theorem, suppose some user u has not decided on a server by time T. We construct a "witness tree[2]" of degree $\kappa + 1$ and depth at least ρ, where $\rho = T/\delta = \kappa \log\log n / \log(\kappa + 1)$. Each node of the witness tree is a server. Each edge of the witness tree is a user whose two nodes correspond to the two servers chosen by that user. We show that the existence of an undecided user in time step T is unlikely by enumerating all possible witness trees and showing that the occurrence of any such witness tree is unlikely. The proof proceeds in the following three steps.

(1) Constructing a witness tree. If algorithm MaxHitRate has not converged to the optimal state at time T, then there exists a user (say u_1) and a server s such that $H_\tau(u_1, s, T) < 100\%$, since user u_1 has not yet found a server with a 100% hit rate. We make server s the root of the witness tree.

We find children for the root s to extend the witness tree as follows. Since $H_\tau(u_1, s, T) < 100\%$, by Lemma 2 we know server s is overbooked at time $t' = t - \delta$, i.e., there are at least $\kappa + 1$ users requesting server s for $\kappa + 1$ distinct applications at time t'. Let $u_1, \ldots, u_{\kappa+1}$ be the users who sent requests to server s at time t'. Wlog, assume that the users $\{u_i\}$ are ordered in ascending order of their IDs. By Lemma 1, we know that the probability of a user deciding on an overbooked server is small, i.e., at most $1/n^{\Omega(1)}$. Thus, with high probability, users $u_1, \ldots, u_{\kappa+1}$ are undecided at time t' since server s is *overbooked*. Let s_i be the other server choice associated with user u_i (one of the choices is server s). We extend the witness tree by creating $\kappa + 1$

[2]A witness tree is so called as it bears witness to the occurrence of an event such as a user being undecided.

children for the root s, one corresponding to each server s_i. Note that for each of the servers s_i we know that $H(u_i, s_i, t') < 100\%$, since otherwise user u_i would have decided on server s_i in time step t'. Thus, analogous to how we found children for s, we can recursively find $\kappa + 1$ children for each of the servers s_i and grow the witness tree to an additional level.

Observe that to add an additional level of the witness tree we went from server s at time T to servers s_i at time t', i.e., we went back in time by an amount of $T - t' \leq \delta$. If we continue the same process, we can construct a witness tree that is a $(\kappa + 1)$-ary tree of depth $T/\delta = \rho$.

(2) Pruning the witness tree. If the nodes of the witness tree are guaranteed to represent distinct servers, proving our probabilistic bound is relatively easy. The reason is that if the servers are unique then the users that represent edges of the tree are unique as well. Therefore the probabilistic choices that each user makes is independent, making it easy to evaluate the probability of occurrence of the tree. However, it may not be the case that the servers in the witness tree constructed above are unique, leading to dependent choices that are hard to resolve. Thus, we create a *pruned witness tree* by removing repeated servers from the original (unpruned) witness tree.

We prune the witness tree by visiting the nodes of the witness tree iteratively in breadth-first search order starting at the root. As we perform breadth-first search (BFS), we remove (i.e., prune) some nodes of the tree and the subtrees rooted at these nodes. What is left after this process is the pruned witness tree. We start by visiting the root. In each iteration, we visit the next node v in BFS order that has not been pruned. Let $\beta(v)$ denote the nodes visited *before* v. If v represents a server that is different from the servers represented by nodes in $\beta(v)$, we do nothing. Otherwise, prune all nodes in the subtree rooted at v. Then, mark the edge from v to its parent as a *pruning edge*. (Note that the pruning edges are not part of the

pruned witness tree.) The procedure continues until either no more nodes remain to be visited or there are $\kappa + 1$ pruning edges. In the latter case, we apply a final pruning by removing all nodes that are yet to be visited, though this step does not produce any more pruning edges. This process results in a pruned witness and a set of p (say) pruning edges.

Note that each pruning edge corresponds to a user who we will call a *pruned user*. We now make a pass through the pruning edges to select a set P of unique pruned users. Initially, P is set to $\emptyset$. We visit the pruning edges in BFS order and for each pruning edge (u, v) we add the user corresponding to (u, v) to P, if this user is distinct from all users currently in P and if $|P| < \lceil p/2 \rceil$, where p is the total number of pruning edges. We stop adding pruned users to set P when we have exactly $\lceil p/2 \rceil$ users. Note that since a user who made server choices of u and v can appear at most twice as a pruned edge, once with u in the pruned witness tree and once with v in the pruned witness tree. Thus, we are guaranteed to find $\lceil p/2 \rceil$ distinct pruned users.

After the pruning process, we are left with a pruned witness tree with nodes representing distinct servers and edges representing distinct users. In addition, we have a set P of $\lceil p/2 \rceil$ distinct pruned users, where p is the number of pruning edges.

(3) Bounding the probability of pruned witness trees. We enumerate possible witness trees and bound their probability using the union bound. Observe that since the (unpruned) witness tree is a $(\kappa + 1)$-ary tree of depth ρ, the number of nodes in the witness tree is

$$m = \sum_{0 \le i \le \rho} (\kappa + 1)^i = \frac{(\kappa + 1)^{\rho+1} - 1}{\kappa} \le 2 \log^2 n, \tag{3.4}$$

since $\rho = 2 \log \log n / \log(\kappa + 1)$ and hence $(\kappa + 1)^\rho = \log^2 n$.

Ways of choosing the shape of the pruned witness tree: The shape of the pruned witness tree is determined by choosing the p pruning edges of the tree. The number of ways of selecting the p pruning edges is at most $\binom{m}{p} \le m^p$, since there are at most

m edges in the (unpruned) witness tree.

Ways of choosing users and servers for the nodes and edges of the pruned witness tree: The enumeration proceeds by considering the nodes in BFS order. The number of ways of choosing the server associated with the root is n. Consider the i^{th} internal node v_i of the pruned witness tree whose server has already been chosen to be s_i. Let v_i have μ_i children. There are at most $\binom{n}{\mu_i}$ ways of choosing distinct servers for each of the μ_i children of v_i. Also, since there are at most n users in the system at any point in time, the number of ways to choose distinct users for the μ_i edges incident on v_i is also at most $\binom{n}{\mu_i}$. There are $\mu_i!$ ways of pairing the users and the servers. Further, the probability that a chosen user chooses server s_i corresponding to node v_i and a specific one of μ_i servers chosen above for v_i's children is

$$\frac{1}{\binom{n}{2}} = \frac{2}{n(n-1)},$$

since each set of two servers is equally likely to be chosen in step 1 of the algorithm. Further, note that each of the μ_i users chose μ_i distinct applications and let the probability of occurrence of this event be $Uniq(n_a, \mu_i)$. This uniqueness probability has been studied in the context of collision-resistant hashing and it is known [5] that $Uniq(n_a, \mu_i)$ is largest when the content popularity distribution is the uniform distribution ($\alpha = 0$) and progressively becomes smaller as α increases. In particular, $Uniq(n_a, \mu_i) \leq e^{-\Theta(\mu_i^2/n_a)} < 1$. Putting it together, the number of ways of choosing a distinct server for each of the μ_i children of v_i, choosing a distinct user for each of the μ_i edges incident on v_i, choosing a distinct application for each user, and multiplying by the appropriate probability is at most

$$\binom{n}{\mu_i} \cdot \binom{n}{\mu_i} \cdot \mu_i! \cdot \left(\frac{2}{n(n-1)}\right)^{\mu_i} \cdot Uniq(n_a, \mu_i) \leq \frac{2^{\mu_i}}{\mu_i!}, \tag{3.5}$$

provided $\mu_i > 1$. Let m' be the number of internal nodes v_i in the pruned witness tree such that $\mu_i = \kappa + 1$. Using the bound in Equation 3.5 for only these m' nodes, the number of ways of choosing the users and servers for the nodes and edges respectively of the pruned witness tree weighted by the probability that these choices occurred is at most

$$n \cdot (2^{\kappa+1}/(\kappa+1)!)^{m'}.$$

Ways of choosing the pruned users in P: Recall that there are $\lceil p/2 \rceil$ distinct pruned users in P. The number of ways of choosing the users in P is at most $n^{\lceil p/2 \rceil}$, since at any time step there are at most n users in the system to choose from. Note that a pruned user has both of its server choices in the pruned witness tree. Therefore, the probability that a given user is a pruned user is at most m^2/n^2. Thus the number of choices for the $\lceil p/2 \rceil$ pruned users in P weighted by the probability that these pruned users occurred is at most

$$n^{\lceil p/2 \rceil} \cdot (m^2/n^2)^{\lceil p/2 \rceil} \leq (m^2/n)^{\lceil p/2 \rceil}.$$

Bringing it all together: The probability that there exists a pruned witness tree with p pruning edges, and m' internal nodes with $(\kappa+1)$ children each, is at most

$$\begin{aligned} & m^p \cdot n \cdot (2^{\kappa+1}/(\kappa+1)!)^{m'} \cdot (m^2/n)^{\lceil p/2 \rceil} \\ & \leq n \cdot (2^{\kappa+1}/(\kappa+1)!)^{m'} \cdot (m^4/n)^{\lceil p/2 \rceil} \\ & \leq n \cdot (2e/(\kappa+1))^{m'(\kappa+1)} \cdot (m^4/n)^{\lceil p/2 \rceil}, \end{aligned} \tag{3.6}$$

since $(\kappa+1)! \geq ((\kappa+1)/e)^{\kappa+1}$. There are two possible cases depending on how the pruning process terminates. If the number of pruning edges, p, equals $\kappa+1$ then the third term of Equation 3.6 is

$$(m^4/n)^{\lceil p/2 \rceil} \leq (16 \log^8 n/n)^{\lceil (\kappa+1)/2 \rceil} \leq 1/n^{\Omega(1)},$$

using Equation 3.4 and assuming that cache size κ is at least a suitably large constant. Alternately, if the pruning process terminates with fewer than $\kappa + 1$ pruning edges, it must be that at least one of the $\kappa + 1$ subtrees rooted at the children of the root s of the (unpruned) witness tree has no pruning edge. Thus, the number of internal nodes m' of the pruned witness tree with $(\kappa+1)$ children each is bounded as follows:

$$m' = \sum_{0 \le i < \rho - 1} (\kappa + 1)^i \ge (\kappa + 1)^{\rho - 2} \ge \log^2 n/(\kappa + 1)^2,$$

as $(\kappa + 1)^\rho = \log^2 n$. Thus, the second term of Equation 3.6 is

$$(2e/(\kappa + 1))^{m'(\kappa+1)} \le (2e/(\kappa + 1))^{\log^2 n/(\kappa+1)} \le 1/n^{\Omega(1)},$$

assuming $\kappa > 2e - 1$ but is at most $O(\log n/ \log \log n)$. Thus, in either case, the bound in Equation 3.6 is $1/n^{\Omega(1)}$. Further, since there are at most m values for p, the total probability of a pruned witness tree is at most $m \cdot 1/n^{\Omega(1)}$ which is $1/n^{\Omega(1)}$. This completes the proof of the theorem. □

Are two or more random choices necessary for all users to receive a 100% hit rate? Analogous to the "power of two choices" in the balls-into-bins context [35], we show that two or more choices are required for good performance with the following theorem.

Theorem 4. *For any fixed constants $0 \le \alpha < 1$ and $\kappa \ge 1$, when algorithm GoWithTheWinner uses one random choice for each user ($\sigma = 1$), the minmax hit rate $H(t) = o(1)$, with high probability, i.e., $H(t)$ tends to zero as n tends to infinity, with high probability.*

Proof. From the classical analysis of throwing n balls into n bins [35], we know that there exist a subset $U' \subseteq U$ such that $|U'| = \Theta(\log n/ \log \log n)$ and all users in U'

have chosen a single server s, with high probability. Now we show that some user in U' must have a small hit rate with high probability. Let C' represent the set of all objects accessed by all users in S'. The probability that $|C'| \leq \kappa w(n)$ can be upper bounded as follows, where $w(n)$ is an arbitrarily slowly growing function of n. The number of ways of picking C' objects from a set C of n objects is at most $n^{|C'|}$. The probability that a user in U' will pick an object in C' can be upper bounded by the probability that a user chooses one of the $|C'|$ most popular objects. Thus the probability that a user in U' picks an object in C' is at most $\mathcal{H}(|C'|, \alpha)/\mathcal{H}(n, \alpha) = \Theta((|C'|/n)^{1-\alpha})$, where $\mathcal{H}(i, \alpha)$ is the i^{th} generalized harmonic number and $\mathcal{H}(i, \alpha) = \Theta(i^{1-\alpha})$.Thus, the probability that all users in U' pick objects in C' is at most $\Theta((|C'|/n)^{(1-\alpha)|U'|})$. Therefore, the probability that $|C'| \leq \kappa w(n)$ is at most

$$\begin{aligned} & n^{|C'|} \cdot \Theta((|C'|/n)^{(1-\alpha)|U'|}) \\ \leq \quad & n^{\kappa w(n)} \cdot (\kappa w(n)/n)^{\Theta(\log n/\log\log n)} = o(1) \end{aligned}$$

Thus, probability that $|C'| \leq \kappa w(n)$ is small and hence $|C'| > \kappa w(n)$, with high probability. Since the minmax hit rate $H(t)$ is at most $\kappa/|C'|$ which is at most $1/w(n)$, $H(t)$ tends to zero with high probability. □

3.2.4 When $n_u = n_s^\alpha, \alpha > 1$

Now we analyze the case that there are many more users than the number of servers. Assume $n_s = n, n_u = n^\alpha$ and $\kappa = \frac{n_u}{n_s} = n^{\alpha-1}$, we have the following result,

Theorem 5. *When $n_s = n, n_u = n^\alpha, \alpha > 1$, with probability at least $1 - \frac{1}{n^{\Omega(1)}}$, the maximum load (number of incoming servers) over all servers is $O(\sigma\frac{n_u}{n_s})$. Furthermore, if $\kappa = \frac{n_u}{n_s}$, all users have 100% hit rate.*

Proof. We prove the Theorem 5 using Chernoff Bound.

We firstly look at the load of one server. Let X_i be the indicator that user u_i selected

the server we looked at, and let $Y = \sum_{i=1}^{n_u} X_i$ be the total number of incoming users at this server. Because each user chooses σ servers uniformly at random, we have $\mathbb{P}(X_i = 1) = \sigma/n_s$. Thus we have,

$$\begin{aligned}
\mathbb{P}\left(Y > \sigma\frac{n_u}{n_s}\right) &= \mathbb{P}\left(Y > \sigma \log n_s\right) \\
&= \mathbb{P}\left(Y \geq \sigma \log n_s + 1\right) \\
&= \mathbb{P}\left(Y \geq (1 + \frac{1}{\sigma \log n_s})\sigma \log n_s\right) \\
&\leq e^{-n_s \log n_s \frac{1}{3(\sigma \log n_s)^2}} \\
&= n_s^{-\frac{n_s}{3\sigma^2 \log^2(n_s)}}.
\end{aligned}$$

With which we can then calculate the bound on the maximum load with union bound. Let Y_i be the number of users at server u_i, then we have

$$\begin{aligned}
\mathbb{P}\left(\max_i\{Y_i\} > \sigma\frac{n_u}{n_s}\right) &= \mathbb{P}\left(\cup_{i=1}^{n_s}\{Y_i > \sigma\frac{n_u}{n_s}\}\right) \\
&\leq \sum_{i=1}^{n_s} \mathbb{P}\left(Y_i > \sigma\frac{n_u}{n_s}\right) \\
&\leq n_s \times n_s^{-\frac{n_s}{3\sigma^2 \log^2(n_s)}} \\
&= n_s^{-(\frac{n_s}{3\sigma^2 \log^2(n_s)} - 1)},
\end{aligned}$$

which is in the order of $\frac{1}{n_s^{\Omega(1)}}$. □

Theorem 5 implies that when $n_u = n_s^\alpha$ all the servers have balanced load of $\sigma\frac{n_u}{n_s}$, and thus we don't need a server selection mechanism for load balancing other than just letting users randomly choose the server. In this case, it's not beneficial to let users start with more than one randomly selected servers, because with $\sigma = 1$ the load on all servers are balanced already. Thus, as long as we have feasible server capacity $\kappa = \omega(\frac{n_u}{n_s})$, all the users will have enough resources from the server and have 100% hit rate by randomly select one server.

The number of content items n_c here does not affect the result of load balancing. Actually, the result stays the same when $n_c \geq n_u$. When the number of content items is much smaller than number of users, $n_c << n_u$, the cache size can be made smaller because the number of distinct requests at each server becomes smaller.

3.3 Bitrate Maximization for Video Content

In video streaming, a key performance metric is the *bitrate* at which a user can download a video. If the server is unable to provide the required bitrate to the user, the video may frequently freeze resulting in an inferior viewing experience and reduced user engagement [29]. For simplicity, we model the server's bandwidth capacity that is often the critical bottleneck resource, while leaving other factors that could influence video performance such as the server-to-user connection and the server's cache[3] for future work.

3.3.1 Problem formulation

The bitrate required to play a stream without freezes is often the encoded bitrate of the stream. For simplicity, we assume that each user requires a bitrate of 1 unit for playing its video and each server has the capacity to serve an aggregation of κ units. We also assume each server evenly divides its available bitrate capacity among all users streaming videos from it. We assume each user can tell the exact bitrate that it receives from its chosen candidate servers and that this bitrate is used as the performance feedback (cf. Figure 3.1).

Unlike web content delivery, where users make random requests to the same website, we assume that users requesting video streaming maintain persistent connections with the server. We use a discrete time model in this case as compared to the con-

[3]Unlike the web, cache hit rate is a less critical determinant of video performance. Videos are cached in chunks by the server. The next chunk is often prefetched from origin if it is not in cache, even while the current chunk is being played by the user, so as to hide the origin-to-server latency.

tinuous time model for web content delivery. We assume after each time unit that users examine the bit rate provided by each of the available servers and then make decisions according to the performance (measured by bit rate). The goal of each user is to find a server that can provide the required bitrate of 1 unit for viewing the video.

3.3.2 Algorithm MaxBitRate

After each user $u \in U$ has selected a video object $c_u \in C$ using the popularity distribution, Algorithm MaxBitRate described below is executed independently by each user $u \in U$, in discrete time steps.

1. Choose a random subset of candidate servers $S_u \subseteq S$ such that $|S_u| = \sigma$.

2. At each time step $t \geq 0$, do the following:

   (a) Request the video content from all servers $s \in S_u$.

   (b) For each server $s \in S_u$, compute $B(u, s, t) \stackrel{\Delta}{=}$ bitrate provided by server s to user u in the current time step.

   (c) If there exists a server $s \in S_u$ such that $B(u, s, t) = 1$, then *decide* on server s by setting $S_u \leftarrow \{s\}$.

Note that each user executes a simple strategy of trying σ randomly chosen servers initially. Then, using the bitrate received in the current time step as feedback, each user independently narrows it's choice of servers to a single server that provides the required unit bitrate. If multiple servers provide the required bitrate, the user selects one at random. Further, note that a user u downloading from a server s at time t knows immediately whether or not the server is overloaded, since server s is overloaded if user u received a bitrate of less than 1 unit from the server, i.e., $B(u, s, t) < 1$. This is a point of simplification in relation to the complex situation of hit rate maximization where any single cache hit is not indicative of a non-overloaded server and a historical average of hit rates over a large enough time window τ is required

as a probabilistic indicator of server overload. Furthermore, this simplification yields both faster convergence to an optimal state in $T = O(\log\log n/\log(\kappa+1))$ steps and a much simpler proof of convergence.

3.3.3 Analysis of Algorithm MaxBitRate

As before, we rigorously analyze the case where $n_u = n_s = n$. Let the *minmax* bitrate $B(t)$ be the best bitrate obtained by the worst user at time t, i.e.,

$$B(t) \stackrel{\Delta}{=} \min_{u\in U} \max_{s\in S} B(u,s,t).$$

Theorem 6. *When $\sigma \geq 2$, the minmax bitrate converges to $B(t) = 1$ unit, for all $t \geq T$, within time $T = O(\log\log n/\log(\kappa+1))$, with high probability. When $\sigma = 1$ on the other hand, the minmax bitrate $B(t) = O(\kappa\log\log n/\log n)$, with high probability. In particular, when $\sigma = 1$ and the cache size κ is $o(\log n/\log\log n)$, including the case when κ is a fixed constant, $B(t)$ tends to zero as n tends to infinity, with high probability.*

Proof. The proof is similar to that of Theorem 3 in that we create a witness tree, prune it, and then show that a pruned witness tree is unlikely. A server s is said to be overloaded at time t if more than κ users want to download from server s in time step t. We construct a witness tree as follows. If the algorithm MaxBitRate has not converged to the optimal state at time $T = 4\log\log n/\log(\kappa+1)$, then there exists an user (say u_1) and a server s such that $B(u_1, s, T-1) < 1$, since user u_1 has not yet found a server that can provide a bitrate of 1 unit. We make sever s the root of the witness tree. We find children for the root s to extend the witness tree as follows. Since $B(u_1, s, T-1) < 1$, there exists $\kappa+1$ distinct users $u_1, \ldots, u_{\kappa+1}$ who sent a request to server s at time $T-1$. We know that users $u_1, \ldots, u_{\kappa+1}$ are undecided at time $T-1$ since they made a request to a overloaded server s. Let s_i

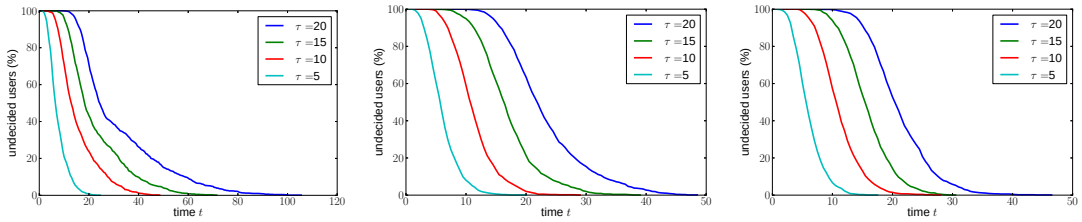
be the other server choice associated with user u_i (one of the choices is server s). We extend the witness tree by creating $\kappa+1$ children for the root s, one corresponding to each server s_i. Note that for each of the servers s_i we know that $B(u_i, s_i, T-2) < 1$, since otherwise user u_i would have decided on server s_i in time step $T-2$. Thus, analogous to how we found children for s, we can recursively find $\kappa+1$ children for each of the servers s_i and grow the witness tree to an additional level. Note that going back two time steps yields an additional level of the witness tree. Thus, we get a witness tree that is a $(\kappa+1)$-ary tree of depth $\rho = T/2 = 2\log\log n/\log(\kappa+1)$. The rest of the proof of pruning and enumerating such witness trees to show that the probability that any such witness tree occurs is at most $1/n^{\Omega(1)}$ is similar to the proof of Theorem 3. □

3.4 Empirical Evaluation

We empirically study our algorithm *GoWithTheWinner* through simulation. Requests from each user is modeled as a Poisson arrival sequence with unit rate. We use $n_u = 1000$ users. To simulate varying numbers of servers, users, and content items, we vary n_s and n_c such that $1 \leq n_u/n_c, n_u/n_s \leq 100$. We also simulate a range of values for the spread $1 \leq \sigma \leq 6$, and sliding window size $1 \leq \tau \leq 20$. Each server implements an LRU replacement policy of size $\kappa \geq 2$. We use the power law distribution for content popularity distribution, where the k^{th} most popular object in C is picked with a probability

$$p_k \triangleq \frac{1}{k^\alpha \cdot \mathcal{H}(n_c, \alpha)}, \tag{3.7}$$

where $\alpha \geq 0$ is the exponent of the distribution and $\mathcal{H}(n_c, \alpha) = \sum_{k=1}^{n_c} 1/k^\alpha$. Note that power law distributions (aka Zipf distributions) are commonly used to model the popularity of online content such as web pages, and videos. This family of distributions is parametrized by a Zipf rank exponent α with $\alpha = 0$ representing the



(a) $\alpha = 0.65$, $n_u/n_s = 1$ (b) $\alpha = 0.65$, $n_u/n_s = 10$ (c) $\alpha = 0.65$, $n_u/n_s = 20$

Figure 3.2: The figures show the percentage of undecided users for a typical power law distribution ($\alpha = 0.65$) with spread $\sigma = 2$ and $n_u = 1000$. Note that the undecided users decrease with time in all cases, but the convergence is faster when we use fewer but larger servers by setting n_u/n_s to be larger. Also, the smaller values of the look-ahead window τ result in faster convergence.

(a) $\alpha = 0.6$, $n_u/n_s = 1$ (b) $\alpha = 0.6$, $n_u/n_s = 20$

Figure 3.3: Generally, as τ increases, convergence time increases but failure rate decreases. It is also true for larger servers ($n_u/n_s = 20$), only the failure has gone to zero for all investigated sliding window sizeτ.

extreme case of an uniform distribution and larger values of α representing a greater skew in the popularity. It has been estimated that the popularity of web content can be modeled by a power law distribution with an α in the range from 0.65 to 0.85 [8, 22, 20]. In the simulations, the content items are requested by users using the power law distribution of (3.7) with $\alpha = 0.65$ to model realistic content popularity [8] [22]. However, we also vary α from 0 (uniform distribution) to 1.5 in some of our simulations.

The system *converges* when all users have decided on a single server from their set of candidate servers. There are two complementary metrics that relate to convergence. *Failure rate* is the probability that the system converged to a non-optimal state where

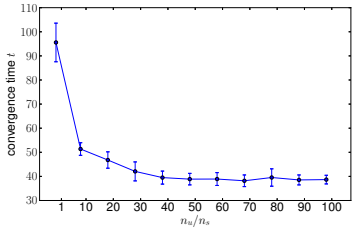


Figure 3.4: As n_u/n_s increases fewer servers with larger capacity are used and convergence time decreases. The decrease is less pronounced beyond $n_u/n_s \geq 40$ under this setting ($\alpha = 0.65$, $\sigma = 2$, $\tau = 20$).

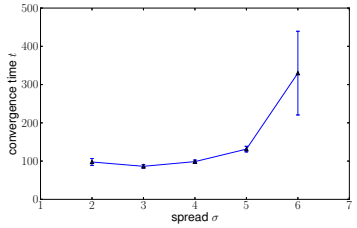


Figure 3.5: There is a very small incremental benefit in using $\sigma = 3$ instead of 2, though higher values of $\sigma > 3$ only increased the convergence time. ($\alpha = 0.65, n_u/n_s = 1, \tau = 20, \kappa = 2.$)

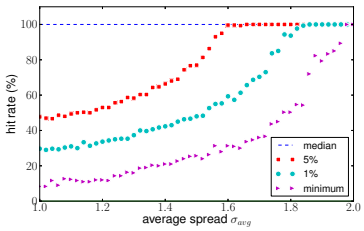


Figure 3.6: Order statistics of the hit rate of the user population. ($\alpha = 0.65, n_u/n_s = 1, \tau = 10, \kappa = 2.$)

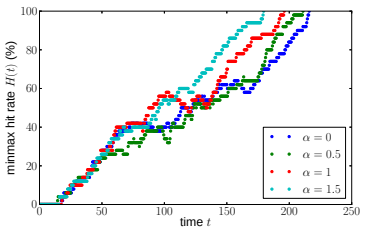


Figure 3.7: Minmax hitrate versus time for different power law distributions.

there exists servers that are overbooked, resulting in some users incurring cache misses after convergence. The failure rate is calculated from multiple runs of the simulation. *Convergence time* is the time it takes for the system to converge provided that it converges to an optimal state.

3.4.1 Speed of convergence

Figure 3.2 shows how the fraction of undecided users decreases over time until it reaches zero, resulting in convergence. Note that users do not decide in the first τ steps, since they must wait at least that long to accumulate a window of τ hits. However, once the first τ steps complete, the decrease in the number of undecided users is fast as users discover that at least one of their two randomly chosen candidate

servers have less load. The rate of decrease in undecided users decreases towards the end, as the number of users who experience cache contention in *both* of their server choices require multiple iterations to resolve.

In this simulation, we keep the number of users $n_u = 1000$ but vary the number of servers n_s to achieve different values for n_u/n_s. Note that for a fair comparison, we keep the system-wide load the same. Load l is a measure of cache contention in the network and is naturally defined as the ratio of the numbers of users in the system and total serving capacity that is available in the system. That is, $l \triangleq n_u/(\kappa \cdot n_s)$. For all three setting of Figure 3.2, we maintain a load $l = 0.5$. The figure shows that with fewer (but larger) servers (n_u/n_s is larger) the convergence time is faster, because having server capacity in a few larger servers provides a larger hit rate than having the same capacity in several smaller servers. Similar performance gains are also found in the context of web caching and parallel jobs scheduling [40]. Convergence times are plotted explicitly in Figure 3.4 for a greater range of user-to-server ratios. As n_u/n_s increases from 1 to 40, convergence time decreases. The decreases in convergence times are not significant beyond $n_u/n_s \geq 40$.

3.4.2 Impact of sliding window τ

The sliding window τ is the number of recent requests used by algorithm *GoWithTheWinner* to estimate the hit rate. As shown in Figure 3.3, there is a natural tradeoff between convergence time and failure rate. When τ increases, the users take longer to converge, as they require a 100% hit rate in a larger sliding window. However, waiting for a longer period also makes their decisions more robust. That is, a user is less likely to choose an overbooked server, since an overbooked server is less likely to provide a string of τ hits for large τ. In our simulations with many smaller caches ($n_u/n_s = 1$), when $\tau \leq 4$, users made quick choices based on a smaller sliding window. But, this resulted in the system converging to a non-optimal state 100% of the time. As τ

further increases, failure rate decreases. The value of $\tau = 11$ is a suitable sweet spot as it results in the smallest convergence time for a zero failure rate. However, for fewer but larger servers ($n_s/n_u = 20$), all selections of window size τ (thus the small values like $\tau = 5$) yielded a 0% failure rate, while convergence time still increases as the window size gets larger.

3.4.3 Impact of spread σ

As shown in Theorems 3 and 4, a spread of $\sigma \geq 2$ is required for the system to converge to an optimal solution, while a spread of $\sigma = 1$ is insufficient. As predicted by our analysis, our simulations did not converge to an optimal state with $\sigma = 1$. Figure 3.5 shows the convergence time as a function of spread, for $\sigma \geq 2$.

As σ increases, there are two opposing factors that impact convergence time. The first factor is that as σ increases, each user has more choices and a user is more likely to find a suitable server with less load. On the other hand, an increase in σ also increases the total number of initial requests in the system that equals σn_u. Thus, the initiate server load increases in σ. These opposing forces result in a very small incremental benefit when using $\sigma = 3$ instead of 2, though the higher values of $\sigma > 3$ showed no benefit as convergence time increases with σ increases.

We established the "power of two random choices" phenomenon where two or more random server choices yield superior results to having just one. It is intriguing to ask *what percentage of users need two choices* to reap the benefits of multiple choices? Consider a mix of users, some with two random choices and others with just one. Let σ_{avg}, $1 \leq \sigma_{avg} \leq 2$, denote the average value of the spread among the users.

In Figure 3.6, we show different order statistics of the hit rate as a function of σ_{avg}. Specifically, we plot the minimum value, 1^{st}-percentile, 5^{th}- percentile and the median (50^{th}-percentile) of user hit rates after simulating the system for 200 time units. As our theory predicts, when $\sigma_{avg} = 2$, the minimum and all the order statistics converge

to 100%, as all users converge to a 100% hit rate. Further, if we are interested in only the median user, any value of the spread is sufficient to guarantee that 50% of the users obtain a 100% hit rate. Perhaps the most interesting phenomena is that if $\sigma_{avg} = 1.7$, i.e., 70% of the users have two choices and the rest have one choice, the 5^{th}-percentile converges to 100%, i.e., all but 5% of the users experience a 100% hit rate. For a higher value of $\sigma_{avg} = 1.9$, the 1^{st}-percentile converges to 100%, i.e., all but the 1% of the users experience a 100% hit rate. This result shows that our algorithm still provides benefits even if only *some* users have multiple random choices of servers available to them.

3.4.4 Impact of demand distribution

We now study how hit rate changes with the exponent α in the power law distribution of Equation 3.7. Note that the distribution is uniform when $\alpha = 0$ and is the harmonic distribution when $\alpha = 1$. As α increases, since the tails fall as a power of α, the distribution becomes more skewed towards content items with a smaller rank. In Figure 5.4, we plot the minmax hitrate over time for different α, where we see that a larger α leads to faster convergence. The reason is that as the popularity distribution gets more skewed, a larger fraction of users will request the same popular content items, leading to higher hit rate and faster convergence. Thus, the uniform popularity distribution ($\alpha = 0$) is the worst case and the algorithm converges faster for the distributions that tend to occur more commonly in practice. Providing theoretical support for this empirical result by analyzing the convergence time to show faster convergence for larger α is a topic for future work.

3.5 Related work

Server selection algorithms have a rich history of both research and actual implementations over the past two decades. Several server selection algorithms have

been proposed and empirically evaluated, including client-side algorithms that use historical performance feedback using probes [19, 15]. Server selection has also been studied in a variety of contexts, such as the web [15, 45], video streaming [48], and cloud services [50]. Our work is distinguished from the prior literature in that we theoretically model the "Go-With-The-Winner" paradigm that is common to many proposed and implemented client-side server selection algorithms. Our work is the first formal study of the efficacy and convergence of such algorithms.

In terms of analytical techniques, our work is closely related to prior work on balls-into-bins games where the witness tree technique was first utilized [35]. Witness trees were subsequently used to analyze load balancing algorithms, and circuit-switching algorithms [14]. However, our setting involves additional complexity requiring novel analysis due to the fact that users can share a single cached copy of an object and the hitrate feedback is only a probabilistic indicator of server overload. Also, our work shows that the "power of two random choices" phenomenon applies in the context of content delivery, a phenomenon known to hold in other contexts such as balls-into-bins, load balancing [52], relay allocation for services like Skype [38], and circuit switching in interconnection networks [35].

3.6 Conclusion

Our work constitutes the first formal study of the simple "Go-With-The-Winner" paradigm in the context of web and video content delivery. For web (resp., video) delivery, we proposed a simple algorithm where each user randomly chooses two or more candidate servers and selects the server that provided the best hit rate (resp., bitrate). We proved that the algorithm converges quickly to an optimal state where all users receive the best hit rate (resp., bitrate) and no server is overloaded, with high probability. While we make some assumptions to simplify the theoretical analysis, our simulations evaluate a broader setting that incorporates a range of values for τ

and σ, varying content popularity distributions, differing load conditions, and situations where only some users have multiple server choices. Taken together, our work establishes that the simple "Go-With-The-Winner" paradigm can provide algorithms that converge quickly to an optimal solution, given a sufficient number of random choices and a sufficiently (but not perfectly) accurate performance feedback.

CHAPTER 4

LINK METRIC TOMOGRAPHY WITH DESIGNED EXPERIMENTS

4.1 Introduction

Network management in a complex network (e.g., MANET-cellular hybrid network, coalition network) often suffers from inefficiencies imposed by protocol/policy barriers between different administration domains, where one notable example is the lack of common monitoring services that provide global state of all the networked components (e.g., links). This limitation motivates the need for external approaches that allow one domain to infer the *internal* state (e.g., link loss rates) of another domain by measuring its *external* performance (e.g., end-to-end losses between a set of vantage points). The methodology of such inference is called *network tomography* [13].

Network tomography has been an active research area in the recent past. Compared with the approach of directly measuring the performance at individual network components, it provides an alternative approach that does not require privileged access to the components. The challenge is that since the measurements are generally functions of the states of multiple components, one has to "invert" these functions. Moreover, the states of interest are usually persistent performance indicators such as mean delays and packet loss rates, while the measurements are functions of the delay/loss instances, and thus the "inversion" has to be robust against randomness in the measurements.

By modeling the performance of each link as a random variable with a (partially) unknown probability distribution, one can apply statistical techniques to estimate

the parameter of this distribution from path measurements [11, 18, 32]. While most existing works focus on estimator design, the accuracy of estimation is fundamentally bounded by the amount of "information" contained in measurements. It is crucial that the probing experiments generate the most informative measurements for estimating link parameters.

It is not straightforward to quantify the information in measurements. On one hand, measurements from different paths provide different amounts of information as they traverse different sets of links, each exhibiting a different level of uncertainty; on the other hand, measurements from a single (multi-hop) path alone do not provide sufficient information for uniquely determining link performances. To address this issue, we apply a notion from estimation theory called the *Fisher Information Matrix (FIM)* [41]. The FIM combines knowledge of paths and link parameters into a single measure of how much "information" a measurement provides for the parameter of interest. By the *Cramér-Rao bound (CRB)* [41], the inverse of the FIM establishes a lower bound on the error covariance matrix of any unbiased estimator.

Based on the FIM, an intuitive formulation of experiment design is to allocate probes on paths to maximize the total information. Turning this intuition into a precise formulation requires an objective function that maps a matrix (FIM) to a scalar that can be uniquely optimized. The theory of optimal experiment design [4] has established a set of such objective functions. In particular, maximizing the determinant of FIM (aka *D-optimality*) leads to a design that minimizes (a bound on) the volume of the error ellipsoid, and minimizing the trace of the inverse FIM (aka *A-optimality*) leads to a design that minimizes (a bound on) the average *mean squared error (MSE)*. Both objective functions lead to convex optimization problems that can, in theory, be solved to obtain the optimal experiment design [7]. Solving these optimizations for practical networks, however, is highly nontrivial, as its solution space (i.e., all possible probe allocations) has a dimension that is at least the size of

the network. In this work, we develop efficient solutions for tomographic experiment design using the above objective functions, and apply our results to two concrete tomography problems with multiplicative/additive link metrics.

4.1.1 Related Work

Based on the model of link metrics, existing work can be classified into algebraic tomography and statistical tomography. Algebraic tomography models each link metric as an unknown constant (e.g., mean link delay) and each path measurement as a deterministic function of link metrics (e.g., mean path delay). The goal of experiment design for algebraic tomography is to construct paths whose measurements can uniquely determine link metrics, e.g., n linearly independent paths for an n-link network [34].

Statistical tomography models each link metric as a random variable with a (partially) unknown probability distribution, and applies various estimation techniques to infer the distribution from path measurements. When multicast is supported, techniques have been proposed to estimate link loss rates and delays from multicast losses and delays [11, 18, 32]. Similar results have been obtained for multi-source measurements [28]. Variants based on subtree, unicast, and striped unicast have also been developed to improve the flexibility of probing [46, 17].

Most existing work in statistical tomography has focused on developing estimators, while the problem of experiment design is often ignored. Unlike algebraic tomography where it suffices to find paths that result in a unique solution of link metrics, statistical tomography also needs to deal with randomness in link metrics and possibly measurement noise. There has been a rich theory on experiment design for general statistical inference, which casts the problem as an optimization of a set of design objectives that capture various aspects of estimation accuracy [4]. The approach has recently been adopted to design experiments for network monitoring, where the prin-

ciple of optimal experiment design has been applied to design link sampling rates for tracking volumes of flows going through the links [47], or design probing sequences for estimating link delays from correlated delays of back-to-back probes [23]. In particular, [23] measures the quality of a probing sequence by the FIM and tries to design probing sequences such that the trace of the inverse FIM can be optimized (i.e., A-optimal design). Their solution, however, relies on a coarse approximation that ignores off-diagonal elements of the FIM. We have a similar goal of designing the optimal allocation of probes based on certain functions of the FIM that capture the overall estimation accuracy, but we identify special structures of the objective functions that allow for exact, closed-form solutions.

4.1.2 Summary of Contributions

Given a number of probes and a set of measurement paths, we want to allocate the probes onto the paths so that the measurements can provide the most accurate estimate of the link parameters of interest. Our specific contributions are:

1. We propose a general experiment design framework for network tomography, where we use the FIM to allocate probes across paths for inferring link parameters using path measurements, with illustrative applications to loss and packet delay variation (PDV) tomography.

2. We derive closed-form estimators for loss and PDV tomography, which, in conjunction with the optimal experiment design, provide asymptotically optimal estimates of the link parameters of interest.

3. For two well-adopted design criteria, *D-optimality* and *A-optimality*, we derive explicit formulas of the design objectives as functions of probe allocation. We also propose a novel criterion, *weighted A-optimality*, that extends A-optimality to incorporate heterogeneity in importance of links. We show how to evaluate these formulas in closed form for loss and PDV tomography.

4. Based on the derived formulas, we develop closed-form solutions of optimal probe allocation when measurement paths form a basis of the vector space of all links. In particular, our solutions show that the D-optimal design leads to a uniform allocation of probes, while the A-optimal design is generally non-uniform. When extra paths are available, we propose a two-step heuristic that first selects a proper basis and then optimizes probe allocation over the basis. Compared with numerical optimization, the proposed solutions significantly improve scalability wrt the number of paths.

5. Observing the dependency of the optimal design on the unknown link parameters, we propose an iterative algorithm that periodically updates the design using refined estimates of the parameters. We show that the result converges to a design based on the true parameters with high probability.

6. We evaluate the proposed designs on real network datasets for both loss and PDV tomography. Our results show that the proposed design based on the A-optimal criterion can effectively reduce the MSE compared with uniform probing, even when the CRB is loose.

The rest of the chapter is organized as follows. Section 4.2 formulates the problem of experiment design for network tomography. Section 4.3 introduces the FIM and its basic properties. Section 4.4 presents estimators of link parameters. Section 4.5 defines objectives of experiment design, and Section 4.6 presents algorithms to optimize the objectives. Section 4.7 evaluates the proposed design algorithms via simulations based on real data. Then Section 4.8 concludes the chapter.

4.2 Problem Formulation

4.2.1 Network Model

Let $\mathcal{G} = (V,\ L)$ denote a network with nodes V and links L. Each link $l \in L$ is associated with a performance metric (e.g., delay, loss) that varies stochastically according to a distribution with unknown parameter θ_l. Let P be a given set of candidate probing paths in $\mathcal{G}$. Each path $p_y \in P$ consists of one or more pairwise adjacent links in[1] $\mathcal{G}$. We assume that the monitoring system can inject probes on all paths in P and observe their end-to-end performance. We also introduce a $|P| \times |L|$ matrix $A := [A_{y,l}]$ defined by P, called the *measurement matrix*, where $A_{y,l} = 1$ if link l is on path p_y and $A_{y,l} = 0$ otherwise. Without loss of generality, we assume that each link is on at least one path in P. Additional assumptions on A (and hence P) will be introduced later as needed. At run time, probes are injected on paths selected according to our experiment design. We consider a probabilistic design model, where each probe is sent over a path randomly selected from P, with probability ϕ_y of selecting path p_y. Here $\boldsymbol{\phi} := (\phi_y)_{y=1}^{|P|}$, satisfying $\phi_y \geq 0$ and $\sum_{y=1}^{|P|} \phi_y = 1$, is a design parameter.

4.2.2 Stochastic Link Metric Tomography

Given a family of link metric distributions with unknown parameters $\boldsymbol{\theta} := (\theta_l)_{l \in L}$, the goal of (parametric) stochastic link metric tomography is to infer $\boldsymbol{\theta}$ from observations of the corresponding performance metrics over probed paths. Let $f_y(x; \boldsymbol{\theta})$ denote the conditional probability of observing path metric x, given that the probe is sent on path p_y and the link parameters are $\boldsymbol{\theta}$. Then the problem of stochastic link metric tomography is to infer the parameter $\boldsymbol{\theta}$ from the observations $(\mathbf{x},\ \mathbf{y}) := (x_t,\ y_t)_{t=1}^{N}$, where x_t is the outcome of the t-th probe and y_t the corresponding path index. Under the assumption that the performance experienced by probes is independent both

[1] We do not impose constraints on paths except that a path traverses each link at most once.

across probes and across links, the observations are i.i.d., each with the following distribution:

$$f(x,\ y;\ \boldsymbol{\theta},\ \boldsymbol{\phi}) = \phi_y f_y(x;\ \boldsymbol{\theta}). \tag{4.1}$$

As concrete examples, we will address in detail two representative performance metrics as follows.

4.2.2.1 Packet Loss Tomography

Packet loss is a typical performance metric that is multiplicative over links on a path. Packet loss tomography aims to infer loss rates on individual links by observing end-to-end packet losses on probed paths. Let the parameter of interest $\boldsymbol{\theta}$ be the vector of link success rates (i.e., complements of loss rates), and each probe outcome x be an indicator that the probe successfully reaches its destination. Assume that losses of the same probe on different links and of different probes on the same link are both independent. Then the observation model becomes:

$$f(x,\ y;\ \boldsymbol{\theta},\ \boldsymbol{\phi}) = \phi_y(\prod_{l \in p_y} \theta_l)^x (1 - \prod_{l \in p_y} \theta_l)^{1-x}. \tag{4.2}$$

4.2.2.2 Packet Delay Variation Tomography

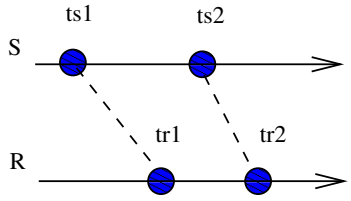


Figure 4.1: Illustration of PDV: t_i^s (t_i^r) is the timestamp of the i-th packet at the sender (receiver).

Packet delay variation (PDV), aka *delay jitter*, is a typical performance metric that is additive over links on a path[2]. PDV between a sender and receiver pair is defined as the difference in sender-to-receiver delays between different packets, i.e., as illustrated in Fig. 4.1, PDV $= (t_2^r - t_2^s) - (t_1^r - t_1^s)$; equivalently, it is the difference between the inter-packet delays at the sender and the receiver, i.e., PDV $= (t_2^r - t_1^r) - (t_2^s - t_1^s)$. The latter definition has the advantage that its evaluation does not require clock synchronization across nodes (assuming the difference in clock speeds is negligible). One can verify that the end-to-end PDV on a path equals the sum of the PDVs at each link. Suppose that PDVs on individual links follow the normal distribution $\mathcal{N}(0,\, \theta_l)$ with zero mean and *unknown* variance θ_l ($l \in L$), and that PDVs experienced by the same probe on different links and by different probes on the same link are both independent. PDV tomography aims to infer $\boldsymbol{\theta}$ from the observed end-to-end PDV x based on the following observation model:

$$f(x,\, y;\, \boldsymbol{\theta},\, \boldsymbol{\phi}) = \phi_y \frac{1}{\sqrt{2\pi \sum_{l \in p_y} \theta_l}} \exp\left(-\frac{x^2}{2 \sum_{l \in p_y} \theta_l} \right). \tag{4.3}$$

4.2.3 Main Problem: Experiment Design

Our goal is to develop a systematic framework to optimally allocate probes over measurement paths such that the overall error in estimating $\boldsymbol{\theta}$ is minimized. Specifically, given an error measure $C(\hat{\boldsymbol{\theta}},\, \boldsymbol{\theta})$ (e.g., L_2-norm) and a total number of probes N, we want to design the probe distribution $\boldsymbol{\phi}$, such that in conjunction with an appropriate estimator $\hat{\boldsymbol{\theta}}$, the expected error $\mathbb{E}[C(\hat{\boldsymbol{\theta}},\, \boldsymbol{\theta})]$ after making N probes is minimized. The specific form of $C(\hat{\boldsymbol{\theta}},\, \boldsymbol{\theta})$ will determine the design criterion, and will be specified later (Section 4.5).

---

[2]Delay is also a typical additive performance metric, where the parameter of interest is usually the mean link delay. We study PDV instead because it has a greater impact on streaming applications.

4.3 Preliminaries

In preparation for FIM-based experiment design, we first review FIM and its important properties.

4.3.1 FIM and CRB

Given an observation model (4.1), the (per-measurement) FIM wrt $\boldsymbol{\theta}$ is an $|L|\times|L|$ matrix, whose $(i,\ j)$-th entry is defined by

$$\mathbb{E}\left[\left(\frac{\partial}{\partial\theta_i}\log f(x,\ y;\ \boldsymbol{\theta},\ \boldsymbol{\phi})\right)\left(\frac{\partial}{\partial\theta_j}\log f(x,\ y;\ \boldsymbol{\theta},\ \boldsymbol{\phi})\right)\middle|\boldsymbol{\theta},\boldsymbol{\phi}\right]. \tag{4.4}$$

We denote this matrix by $I(\boldsymbol{\theta};\ \boldsymbol{\phi})$ to highlight its dependence on both the (unknown) parameter $\boldsymbol{\theta}$ and the design parameter $\boldsymbol{\phi}$. All subsequent references to "FIM" mean this per-measurement FIM.

The significance of the FIM is that it provides a fundamental bound on the error of unbiased estimators. Specifically, if $\hat{\boldsymbol{\theta}}$ is an unbiased estimator of $\boldsymbol{\theta}$ using N i.i.d. measurements, then the covariance matrix of $\hat{\boldsymbol{\theta}}$ satisfies[3] $\mathrm{cov}(\hat{\boldsymbol{\theta}}) \succeq \frac{1}{N} I^{-1}(\boldsymbol{\theta};\ \boldsymbol{\phi})$, known as the *Cramér-Rao bound (CRB)* [41]. In particular, the MSE in estimating θ_l, given by $\mathrm{cov}(\hat{\boldsymbol{\theta}})_{l,l}$, is lower bounded by $I^{-1}_{l,l}(\boldsymbol{\theta};\ \boldsymbol{\phi})/N$.

4.3.2 Identifiability and Invertibility of FIM

The CRB has an implicit assumption that the FIM is invertible. In our problem, we will show that this assumption follows from the identifiability of link parameters. We say that an unknown parameter $\boldsymbol{\theta}$ is *identifiable* from observations $\mathbf{x}$ if and only if the observation model satisfies that $f(\mathbf{x};\ \boldsymbol{\theta}) \neq f(\mathbf{x};\ \boldsymbol{\theta}')$ at some $\mathbf{x}$ for any $\boldsymbol{\theta} \neq \boldsymbol{\theta}'$. In network tomography, the identifiability of link parameters $\boldsymbol{\theta}$ has direct implication on the measurement matrix and the FIM.

[3]For matrices A and B, $A \succeq B$ means that $A - B$ is positive semi-definite.

Specifically, suppose that a stochastic link metric tomography problem can be cast as a linear system $A\mathbf{z}(\boldsymbol{\theta}) = \mathbf{w}$, where A is the measurement matrix, $\mathbf{z}(\boldsymbol{\theta})$ is a bijection of $\boldsymbol{\theta}$, and $\mathbf{w}$ is a vector of path performance parameters such that the probe outcomes depend on $\boldsymbol{\theta}$ only through $\mathbf{w}$. Suppose that $\mathbf{w}$ can be estimated consistently from probes[4]. Then the following statements hold:

- $\boldsymbol{\theta}$ is identifiable if and only if A has full column rank;
- if $\boldsymbol{\theta}$ is identifiable, then $I(\boldsymbol{\theta};\, \boldsymbol{\phi})$ is invertible.

The first statement can be easily proved by an argument of contradiction, and the second statement is a direct implication of the equivalence between the invertibility of the FIM and the *local identifiability*[5] of $\boldsymbol{\theta}$ [44].

Both loss tomography and PDV tomography admit a linear system model $A\mathbf{z} = \mathbf{w}$, where $z_l = \log \theta_l$, $w_y = \log\left(\prod_{l \in p_y} \theta_l\right)$ for loss tomography, and $z_l = \theta_l$, $w_y = \sum_{l \in p_y} \theta_l$ for PDV tomography (the same applies to delay).

Discussion: The aspect of experiment design focusing on path construction has been extensively studied in the literature. If the routing of probes is controllable (subject to cycle-free constraint), then identifiability can be guaranteed by constructing paths using the *Spanning Tree-based Path Construction (STPC)* algorithm in [34]; if the routing is uncontrollable and the default routes between monitors cannot identify all link parameters, then we can transform the topology into a logical topology as in [53], whose links represent the *Minimal Identifiable Link Sequences (MILS)*, such that parameters of the logical links are identifiable.

In this work, we focus on a different aspect of designing probe allocation. Intuitively, identifiability is a basic requirement for the inference problem to be solvable

---

[4]That is, there exists an estimator $\hat{\mathbf{w}}$ that converges to $\mathbf{w}$ in probability as the number of probes goes to infinity.

[5]That is, there exists an open neighborhood of $\boldsymbol{\theta}$ such that no $\boldsymbol{\theta}'$ ($\boldsymbol{\theta}' \neq \boldsymbol{\theta}$) in this neighborhood leads to the same observation model.

with infinite measurements, and probe allocation further maximizes the inference accuracy with finite measurements. Therefore, we will assume in the sequel that *the link parameters of interest are identifiable* using the given paths. By the above statements, this implies that the measurement matrix A has full column rank and the FIM $I(\boldsymbol{\theta};\, \boldsymbol{\phi})$ is invertible. Conceptually, probe allocation among all possible paths generalizes path construction because it specifies not only which paths are used for probing but also how often each path is probed.

4.3.3 Example

We illustrate FIM-based experiment design by a simple example in Fig. 4.2, where we want to use end-to-end losses on paths p_1, p_2, and p_3 to infer loss rates of links l_1 and l_2. Consider three candidate designs $\boldsymbol{\phi}_1 = (\frac{1}{3},\, \frac{1}{3},\, \frac{1}{3})$, $\boldsymbol{\phi}_2 = (0.5,\, 0.5,\, 0)$, and $\boldsymbol{\phi}_3 = (0.15,\, 0.85,\, 0)$. The average CRB for loss rate estimation, given by the average of diagonal elements of the inverse FIM, equals 0.6, 0.5, and 0.98 respectively for the three designs, if the actual link loss rates are (0.5, 0.5); however, if the link loss rates are (0.99, 0.5), the CRB becomes 0.21, 0.26, and 0.18 respectively (see (5.10) in Section 4.5.4.1 for computation of the FIM and hence the CRB). The example demonstrates that: (i) the usual approach of uniformly allocating probes (i.e., $\boldsymbol{\phi}_1$) is generally suboptimal, and (ii) the optimal allocation depends not only on the paths but also on the link parameters. Moreover, although the preferred design ($\boldsymbol{\phi}_2$ or $\boldsymbol{\phi}_3$) in the above cases does not use p_3, it is not clear whether this is always true, as a measurement on p_3 provides information about both links.

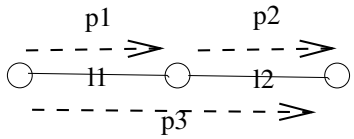


Figure 4.2: Example: loss tomography using three paths.

4.4 Link Parameter Estimation

Fundamental to experiment design is how the collected measurements will be used to estimate the parameters of interest. To this end, we review a well-known estimator and its special relationship with FIM-based experiment design.

4.4.1 Maximum Likelihood Estimator (MLE)

Given observations, the MLE solves for the parameter value that maximizes the likelihood of these observations and uses this value as an estimate of the parameter. The MLE plays a significant role in FIM-based experiment design. Using the FIM in experiment design implies an implicit assumption that the adopted estimator is *efficient* (i.e., unbiased and achieves the CRB), and thus the CRB characterizes estimation error. In this regard, the MLE has a superior property that it is the only candidate for efficient estimator, i.e., if an efficient estimator exists, it must be the MLE [49]. Moreover, although efficient estimators may not exist for finite sample sizes, the MLE is asymptotically efficient under regularity conditions, i.e., its expectation converges to the true parameter at a rate approximating the CRB. Therefore, using the MLE to estimate link parameters guarantees that our FIM-based experiment design will optimize the decaying rate of error as the number of probes becomes large.

4.4.2 MLE for Packet Loss Tomography

The MLE has a unique property that it is *invariant under one-to-one parameter transformations*. That is, if $\hat{\theta}$ is an MLE of θ and $\eta = g(\theta)$ is a one-to-one transformation, then $\hat{\eta} = g(\hat{\theta})$ is an MLE of η. For tomography problems, this property can be leveraged to greatly simplify the derivation of the MLE. Specifically, let $\alpha_y(\boldsymbol{\theta}) := \prod_{l \in p_y} \theta_l$ denote the success probability of path p_y, $n_{1,y}$ the number of successfully received probes, and $n_{0,y}$ the number of lost probes. It is known that the MLE of $\alpha_y(\boldsymbol{\theta})$ is simply the empirical path success probability $\hat{\alpha}_y := n_{1,y}/(n_{1,y}+n_{0,y})$,

as $n_{1,y}$ can be viewed as a sum of $n_{0,y} + n_{1,y}$ i.i.d. Bernoulli random variables with success probability $\alpha_y(\boldsymbol{\theta})$. Moreover, when A has full column rank, the link success rates $\boldsymbol{\theta}$ and the path success rates $\boldsymbol{\alpha} := (\alpha_y(\boldsymbol{\theta}))_{y=1}^{|P|}$ form a one-to-one mapping $\log \boldsymbol{\theta} = (A^T A)^{-1} A^T \log \boldsymbol{\alpha}$ (assume $\boldsymbol{\alpha} > 0$). Using the invariance property of MLE, we can obtain the MLE of $\boldsymbol{\theta}$ from the MLE of $\boldsymbol{\alpha}$ as follows. Without loss of generality, we assume that $n_{1,y} + n_{0,y} > 0$ for $y = 1, \ldots, |P|$.

Proposition 7. *If the measurement matrix A has full column rank and there is at least one successful probe per path (i.e., $n_{1,y} > 0$ for $y = 1, \ldots, |P|$), then the MLE for loss tomography equals*[6]*:*

$$\hat{\boldsymbol{\theta}} = \exp\left((A^T A)^{-1} A^T \log \hat{\boldsymbol{\alpha}}\right), \tag{4.5}$$

where $\hat{\boldsymbol{\alpha}}$ is the vector of empirical path success rates.

Remark: The MLE for loss tomography is only asymptotically unbiased (verified in Section 4.7.2) because of the non-linear operators (log, exp).

Example

Consider a simple 2-link network as illustrated in Fig... Applying the MLE formula in (5.2) yields that the MLE of $(\theta_1,\ \theta_2)^T$ is

$$\begin{cases} \hat{\theta}_1 &= \frac{n_{1,1}}{n_{1,1}+n_{0,1}}, \\ \hat{\theta}_2 &= \frac{n_{1,2}(n_{1,1}+n_{0,1})}{n_{1,1}(n_{1,2}+n_{0,2})}. \end{cases} \tag{4.6}$$

As mentioned before, the MLE is not always efficient. In particular, it is biased in this example as shown below. While $\mathbb{E}[\hat{\theta}_1] = \theta_1(1 - \phi_2^N)$ is not far from the true value θ_1 (assuming $n_{1,1}/(n_{1,1} + n_{0,1}) = 0$ if $n_{1,1} + n_{0,1} = 0$), it can be verified that

[6]For ease of presentation, we use $g(\mathbf{z})$ to denote the vector obtained by applying a scalar function $g(\cdot)$ to each element of a vector $\mathbf{z}$.

$\mathbb{E}[\hat{\theta}_2]$ does not exist, because conditioned on $n_{1,1} + n_{0,1} = n$ for any $0 < n < N$, the conditional expectation

$$\begin{aligned}\mathbb{E}\big[\hat{\theta}_2 | n_{1,1} + n_{0,1} = n\big] =& \frac{n}{N-n}\mathbb{E}\big[n_{1,2} | n_{1,2} + n_{0,2} = N - n\big] \\ & \cdot \mathbb{E}\big[\frac{1}{n_{1,1}} | n_{1,1} + n_{0,1} = n\big] \end{aligned} \tag{4.7}$$

does not exist due to the divergence of the negative moment $\mathbb{E}[\frac{1}{n_{1,1}} | n_{1,1} + n_{0,1} = n]$. Meanwhile, note that the bias will disappear as $N \to \infty$, as the probability for $n_{1,1}$, $n_{1,1} + n_{0,1}$, or $n_{1,2} + n_{0,2}$ to be zero diminishes. We leave detailed study of conditions for the MLE to be efficient for a general network to future work.

4.4.3 MLE for PDV Tomography

We follow a similar approach to derive the MLE for PDV tomography. Specifically, let $\sigma_y(\boldsymbol{\theta}) := \sum_{l \in p_y} \theta_l$ denote the PDV variance on path p_y. Under the zero-mean assumption, it is known that the MLE of $\sigma_y(\boldsymbol{\theta})$ is the empirical path variance $\hat{\sigma}_y := \frac{1}{n_y} \sum_{k=1}^{n_y} x_{y,k}^2$, where n_y is the number of probes sent on path p_y and $x_{y,k}$ the end-to-end PDV for the k-th probe on p_y; this MLE is unbiased. When A has full column rank, the link PDV variances $\boldsymbol{\theta}$ and the path PDV variances $\boldsymbol{\sigma} := (\sigma_y(\boldsymbol{\theta}))_{y=1}^{|P|}$ form a one-to-one transformation $\boldsymbol{\theta} = (A^T A)^{-1} A^T \boldsymbol{\sigma}$. We can then obtain the MLE of $\boldsymbol{\theta}$ as follows (assuming $n_y > 0$ for $y = 1, \ldots, |P|$).

Proposition 8. *If the measurement matrix A has full column rank, then the MLE for PDV tomography equals:*

$$\hat{\boldsymbol{\theta}} = (A^T A)^{-1} A^T \hat{\boldsymbol{\sigma}}, \tag{4.8}$$

where $\hat{\boldsymbol{\sigma}}$ is the vector of empirical path PDV variances.

Remark: The MLE for PDV tomography is unbiased (verified in Section 4.7.3).

Requirements on probing experiment: Applying the MLE formulas in Propositions 18 and 8 imposes certain requirements on the probing experiment: the set of paths for which there is at least one successful probe per path should form a full-column-rank measurement matrix (note that each probe for PDV measurement contains at least two packets). One way to satisfy this requirement is to employ an *initialization phase*, where we send one probe per path (recall that the entire path set P is assumed to give a full-column-rank measurement matrix). In the case of loss tomography, we also need to ensure non-zero empirical path success rates; we find a modified estimate $\widetilde{\alpha}_y = 1/(1 + n_{0,y})$ for paths without a successful probe performs well[7] (note that the error caused by this modification diminishes as the number of probes increases).

4.5 Objective of Experiment Design

The essence of FIM-based experiment design is to treat the CRB as an approximation of the estimation error matrix and select the design parameter $\boldsymbol{\phi}$ to optimize a given objective function based on the CRB. Given an estimate of $\boldsymbol{\theta}$, the FIM (and hence the CRB) only depends on $\boldsymbol{\phi}$, which in theory allows us to optimize $\boldsymbol{\phi}$. Solving this optimization, however, is highly nontrivial as it is an optimization of a $|P|$-dimensional vector, making numerical solutions infeasible for larger $|P|$. In this section, we will show that under certain conditions, satisfied by both loss and PDV tomography, the objective function has a special structure that allows for closed-form solutions.

[7]Alternatively, one may keep probing each path until obtaining a success; this procedure is however not robust for paths with low success rates.

4.5.1 D-Optimal Design

In D-optimal experiment design, we seek to minimize the determinant of the inverse FIM, $\det(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi}))$, or equivalently maximize $\det(I(\boldsymbol{\theta};\boldsymbol{\phi}))$. The CRB implies that this design minimizes the volume of the error ellipsoid.

We begin by establishing a special structure of $\det(I(\boldsymbol{\theta};\boldsymbol{\phi}))$ that holds for any network topology and any set of probing paths, under certain conditions on the observation model. We first show a general property of the FIM as follows.

Lemma 9. *The FIM for the observation model (4.1) is a convex combination of per-path FIMs:*

$$I(\boldsymbol{\theta};\boldsymbol{\phi}) = \sum_{y=1}^{|P|} \phi_y I^{(y)}(\boldsymbol{\theta}), \tag{4.9}$$

where $I^{(y)}(\boldsymbol{\theta})$ is the FIM for path p_y based on the observation model $f_y(x;\,\boldsymbol{\theta})$. Note that $I^{(y)}(\boldsymbol{\theta})$ is independent of $\boldsymbol{\phi}$ and is only a function of $\boldsymbol{\theta}$.

Proof of Lemma 9. Let $\mathcal{L}(\boldsymbol{\theta})$ and $\mathcal{L}_y(\boldsymbol{\theta})$ be the log-likelihood functions for the overall experiment and path p_y, respectively (both are implicitly functions of x and $\boldsymbol{\phi}$). Since $\mathcal{L}(\boldsymbol{\theta}) = \log\phi_y + \mathcal{L}_y(\boldsymbol{\theta})$, applying the definition of FIM in (5.9) yields:

$$I_{i,j}(\boldsymbol{\theta};\boldsymbol{\phi}) = \sum_{y=1}^{|P|} \phi_y \mathbb{E}\left[\left(\frac{\partial}{\partial\theta_i}\mathcal{L}_y(\boldsymbol{\theta})\right)\left(\frac{\partial}{\partial\theta_j}\mathcal{L}_y(\boldsymbol{\theta})\right)\middle|\boldsymbol{\theta},\boldsymbol{\phi},y\right], \tag{4.10}$$

and $\mathbb{E}\left[\left(\frac{\partial\mathcal{L}_y(\boldsymbol{\theta})}{\partial\theta_i}\right)\left(\frac{\partial\mathcal{L}_y(\boldsymbol{\theta})}{\partial\theta_j}\right)\middle|\boldsymbol{\theta},\boldsymbol{\phi},y\right]$ equals $I^{(y)}_{i,j}(\boldsymbol{\theta})$ by definition. □

Based on this decomposition, we can show that the determinant of the FIM has a particular structure as follows.

Theorem 10. *Let $\mathcal{S}_n$ be the collection of all size-n subsets of $\{1,\ldots,|P|\}$. If the per-path FIM satisfies*

$$I^{(y)}_{i,k}(\boldsymbol{\theta})I^{(y)}_{j,l}(\boldsymbol{\theta}) = I^{(y)}_{i,l}(\boldsymbol{\theta})I^{(y)}_{j,k}(\boldsymbol{\theta}) \tag{4.11}$$

for any $y \in \{1,\ldots,|P|\}$ and any $i,j,k,l \in \{1,\ldots,|L|\}$, then there exist functions $B_C(\boldsymbol{\theta})$ ($C \in \mathcal{S}_{|L|}$) such that

$$\det(I(\boldsymbol{\theta};\boldsymbol{\phi})) = \sum_{C\in\mathcal{S}_{|L|}} B_C(\boldsymbol{\theta}) \prod_{i\in C} \phi_i. \tag{4.12}$$

Functions $B_C(\boldsymbol{\theta})$ ($C \in \mathcal{S}_{|L|}$) do not depend on $\boldsymbol{\phi}$.

Proof of Theorem 10. Applying the Leibniz formula for determinant to the decomposed FIM in (4.9) shows that

$$\det(I(\boldsymbol{\theta};\boldsymbol{\phi})) = \sum_{\boldsymbol{\pi}\in\Pi_{|L|}} \operatorname{sgn}(\boldsymbol{\pi}) \prod_{i=1}^{|L|} I_{i,\pi_i}(\boldsymbol{\theta};\boldsymbol{\phi}) \tag{4.13}$$

$$= \sum_{\boldsymbol{\pi}\in\Pi_{|L|}} \operatorname{sgn}(\boldsymbol{\pi}) \left(\sum_{y_1=1}^{|P|} \cdots \sum_{y_{|L|}=1}^{|P|} \prod_{i=1}^{|L|} \phi_{y_i} I^{(y_i)}_{i,\pi_i}(\boldsymbol{\theta}) \right), \tag{4.14}$$

where $\boldsymbol{\pi}$ is a permutation of $\{1,\ldots,|L|\}$ ($\Pi_{|L|}$ is the set of all permutations), and $\operatorname{sgn}(\boldsymbol{\pi})$ is a sign function that equals 1 if $\boldsymbol{\pi}$ is achievable by an even number of pairwise swaps, and -1 if it is achievable by an odd number of swaps. Equation (4.14) shows that the determinant of the FIM can be written as a sum of order-$|L|$ terms of $\boldsymbol{\phi}$ (i.e., $\prod_{i=1}^{|L|}\phi_{y_i}$), weighted by functions of $\boldsymbol{\theta}$. Each term in the summation is uniquely determined by $\boldsymbol{\pi}$ and $\mathbf{y}$.

The key to the proof is to show that after combining these order-$|L|$ terms, the remaining terms only contain product of $|L|$ *distinct* ϕ_y's, i.e., terms containing duplicate variables ($y_i = y_j$ for $i \neq j$) all disappear. We prove this by showing that terms with duplicate variables will combine to zero.

For each term with at least one duplicate variable, i.e., the corresponding $\boldsymbol{\pi}$ and $\mathbf{y}$ satisfy: $\exists i, j \in \{1, \ldots, |L|\}$ $(i \neq j)$ such that $y_i = y_j = y_0$ for some $y_0 \in \{1, \ldots, |P|\}$, there must exist a corresponding term, referred to as the *opposite term*, for the same $\mathbf{y}$ and a slightly different permutation $\boldsymbol{\pi}'$ that is identical as $\boldsymbol{\pi}$ except that $\pi'_i = \pi_j$ and $\pi'_j = \pi_i$. The absolute value of this opposite term equals

$$\left(\prod_{k \neq i,j} \phi_{y_k} I^{(y_k)}_{k,\pi'_k}(\boldsymbol{\theta})\right) \phi^2_{y_0} I^{(y_0)}_{i,\pi'_i}(\boldsymbol{\theta}) I^{(y_0)}_{j,\pi'_j}(\boldsymbol{\theta}),$$

which equals the absolute value of the first term

$$\left(\prod_{k \neq i,j} \phi_{y_k} I^{(y_k)}_{k,\pi_k}(\boldsymbol{\theta})\right) \phi^2_{y_0} I^{(y_0)}_{i,\pi_i}(\boldsymbol{\theta}) I^{(y_0)}_{j,\pi_j}(\boldsymbol{\theta})$$

because $I^{(y_0)}_{i,\pi'_i}(\boldsymbol{\theta}) I^{(y_0)}_{j,\pi'_j}(\boldsymbol{\theta}) = I^{(y_0)}_{i,\pi_i}(\boldsymbol{\theta}) I^{(y_0)}_{j,\pi_j}(\boldsymbol{\theta})$. Meanwhile, $\operatorname{sgn}(\boldsymbol{\pi})$ and $\operatorname{sgn}(\boldsymbol{\pi}')$ must differ as the permutations differ by one pairwise swap. Therefore, the two terms sum up to zero.

Moreover, if we define the *opposite term* of a term containing duplicate variables as the term obtained by swapping π_i and π_j for the first two duplicate variables (i.e., for the smallest i, j with $y_i = y_j$), then it is easy to see that the opposite term of the opposite term is the original term, and thus no two different terms can have the same opposite. Therefore, after combining terms, only terms consisting of a product of $|L|$ distinct ϕ_y's remain, implying formula (4.12). □

Remark: This theorem describes a generic structure of $\det(I(\boldsymbol{\theta}; \boldsymbol{\phi}))$ that applies to any tomography problem where condition (4.11) holds. In words, condition (4.11) states that any 2×2 submatrix of the per-path FIM formed by removing $|L| - 2$ rows and $|L| - 2$ columns has a determinant of zero, i.e., *any* 2×2 *minor of the per-path*

FIM (and the overall FIM) is zero[8] (note that the condition holds trivially if $i = j$ or $k = l$). We will see in Section 4.5.4 that this condition holds for both loss and PDV tomography.

The essence of this theorem is that under condition (4.11), the determinant of the FIM, when viewed as a function of $\boldsymbol{\phi}$, can be written as a weighted sum of order-$|L|$ terms, where each term is a product of $|L|$ (out of $|P|$) *distinct* ϕ_i's. We will show later that this property helps to simplify our FIM-based experiment design.

In fact, analogous arguments can be used to show a formula for any minor of the FIM as follows.

Corollary 11. *Let M be an n-dimensional submatrix of $I(\boldsymbol{\theta}; \boldsymbol{\phi})$ after removing $|L|-n$ rows and columns, and $\mathcal{S}_n$ be defined as in Theorem 10. Then under condition (4.11), there exist functions $B_C(\boldsymbol{\theta})$ ($C \in \mathcal{S}_n$) such that the determinant of M (i.e., a* minor *of $I(\boldsymbol{\theta}; \boldsymbol{\phi})$) equals:*

$$\det(M(\boldsymbol{\theta}; \boldsymbol{\phi})) = \sum_{C \in \mathcal{S}_n} B_C(\boldsymbol{\theta}) \prod_{i \in C} \phi_i. \tag{4.15}$$

Functions $B_C(\boldsymbol{\theta})$ ($C \in \mathcal{S}_n$) do not depend on $\boldsymbol{\phi}$.

4.5.2 A-Optimal Design

In A-optimal experiment design, we seek to minimize the trace of the inverse FIM, $\mathrm{Tr}(I^{-1}(\boldsymbol{\theta}; \boldsymbol{\phi}))$. The CRB states that this design minimizes the average mean squared error (MSE) for estimating $\boldsymbol{\theta}$.

We observe a special structure of $\mathrm{Tr}(I^{-1}(\boldsymbol{\theta}; \boldsymbol{\phi}))$ as follows. Theorem 10 implies, in particular, that when $|P| = |L|$, the determinant of the FIM equals:

---

[8] The $k \times k$ minor of an $m \times n$ matrix is the determinant of a submatrix obtained by removing $m - k$ rows and $n - k$ columns.

$$\det(I(\boldsymbol{\theta};\boldsymbol{\phi})) = B(\boldsymbol{\theta}) \prod_{k=1}^{|L|} \phi_k. \tag{4.16}$$

This fact, together with Corollary 11, can be used to prove the following structure of $\mathrm{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi}))$.

Theorem 12. *Suppose $|P| = |L|$ and the FIM is invertible. If condition (4.11) holds, then the trace of the inverse FIM $\mathrm{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi}))$ admits the following representation:*

$$\mathrm{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})) = \sum_{i=1}^{|L|} \frac{1}{\phi_i} A_i(\boldsymbol{\theta}), \tag{4.17}$$

where $A_1(\boldsymbol{\theta}), \ldots, A_{|L|}(\boldsymbol{\theta})$ are only functions of $\boldsymbol{\theta}$.

Proof of Theorem 12. Let us denote the (i,j)-element of $I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})$ by $I^{-1}_{i,j}(\boldsymbol{\theta};\boldsymbol{\phi})$. Applying Cramer's rule of calculating the inverse of a matrix, we can write

$$I^{-1}_{i,j}(\boldsymbol{\theta};\boldsymbol{\phi}) = (-1)^{i+j} \frac{\det(M_{ji}(\boldsymbol{\theta};\boldsymbol{\phi}))}{\det(I(\boldsymbol{\theta};\boldsymbol{\phi}))}, \tag{4.18}$$

where $\det(M_{ji}(\boldsymbol{\theta};\boldsymbol{\phi}))$ is the minor of element (j,i) of $I(\boldsymbol{\theta};\boldsymbol{\phi})$ (i.e., the determinant of the submatrix after removing row j and column i). In particular, the diagonal elements of $I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})$ have the following form:

$$I^{-1}_{k,k}(\boldsymbol{\theta};\boldsymbol{\phi}) = \frac{\det(M_{kk}(\boldsymbol{\theta};\boldsymbol{\phi}))}{\det(I(\boldsymbol{\theta};\boldsymbol{\phi}))},\ k = 1, \ldots, |L|. \tag{4.19}$$

By Corollary 11, the numerator can be written as:

$$\det(M_{kk}(\boldsymbol{\theta};\boldsymbol{\phi})) = \sum_{C \in \mathcal{S}_{|L|-1}} B_{C,k}(\boldsymbol{\theta}) \prod_{i \in C} \phi_i. \tag{4.20}$$

The trace of $I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})$ is thus equal to

$$\begin{aligned}\mathrm{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})) &= \sum_{k=1}^{|L|} I^{-1}_{k,k}(\boldsymbol{\theta};\boldsymbol{\phi}) \\ &= \sum_{C\in\mathcal{S}_{|L|-1}} \frac{\prod_{i\in C}\phi_i}{\prod_{s=1}^{|L|}\phi_s}\left(\sum_{k=1}^{|L|}\frac{B_{C,k}(\boldsymbol{\theta})}{B(\boldsymbol{\theta})}\right), \end{aligned} \tag{4.21}$$

where we have used the representation (4.16) for $\det(I(\boldsymbol{\theta};\boldsymbol{\phi}))$. Next, we observe that $\mathcal{S}_{|L|-1}$ has exactly $|L|$ members $C_1,\ldots,C_{|L|}$, where each C_i is the subset of $\{1,\ldots,|L|\}$ that excludes i. Thus,

$$\mathrm{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})) = \sum_{i=1}^{|L|}\frac{1}{\phi_i}\left(\sum_{k=1}^{|L|}\frac{B_{C_i,k}(\boldsymbol{\theta})}{B(\boldsymbol{\theta})}\right) = \sum_{i=1}^{|L|}\frac{1}{\phi_i}A_i(\boldsymbol{\theta}),$$

where $A_i(\boldsymbol{\theta}) := \sum_{k=1}^{|L|}\frac{B_{C_i,k}(\boldsymbol{\theta})}{B(\boldsymbol{\theta})}$. □

Remark: The proof actually gives a more general structure of $\mathrm{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi}))$ for any $|P|\geq|L|$:

$$\mathrm{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi}))=\frac{\sum_{C'\in\mathcal{S}_{|L|-1}}\prod_{i\in C'}\phi_i\left[\sum_{k=1}^{|L|}B_{C',k}(\boldsymbol{\theta})\right]}{\sum_{C\in\mathcal{S}_{|L|}}B_C(\boldsymbol{\theta})\prod_{i\in C}\phi_i}, \tag{4.22}$$

where $B_{C',k}(\boldsymbol{\theta})$ and $B_C(\boldsymbol{\theta})$ are only functions of $\boldsymbol{\theta}$. We only highlight the special case of $|P|=|L|$ because it allows for efficient optimization of $\boldsymbol{\phi}$; see Section 4.6.1.

4.5.3 Weighted A-Optimal Design

In practice, applications may place different weights on the links. We extend the A-optimal design to account for this by introducing a *weight vector* $\boldsymbol{\omega} := (\omega_k)_{k=1}^{|L|}$, where ω_k denotes the weight of link l_k. Introducing weights changes the objective from minimizing $\mathrm{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi}))$ to minimizing a *weighted trace* of $I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})$, i.e., the weighted sum of the diagonal elements of $I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})$: $\sum_{k=1}^{|L|}\omega_k I^{-1}_{k,k}(\boldsymbol{\theta};\boldsymbol{\phi})$. By the CRB,

this design minimizes the weighted average MSE for estimating $\{\theta_l\}_{l\in L}$. We refer to this design as the *weighted A-optimal design*.

Using analogous arguments, we can easily extend Theorem 12 to the following result.

Corollary 13. *Under the conditions in Theorem 12, the weighted trace of the inverse FIM admits the following representation:*

$$\sum_{k=1}^{|L|} \omega_k I_{k,k}^{-1}(\boldsymbol{\theta};\boldsymbol{\phi}) = \sum_{i=1}^{|L|} \frac{1}{\phi_i} \widetilde{A}_i(\boldsymbol{\theta}), \tag{4.23}$$

where $\widetilde{A}_1(\boldsymbol{\theta}), \ldots, \widetilde{A}_{|L|}(\boldsymbol{\theta})$ *are only functions of* $\boldsymbol{\theta}$.

Remark: Since the weighted A-optimal design contains the A-optimal design as a special case, we only consider the weighted A-optimal design in the sequel, simply referred to as "A-optimal".

4.5.4 Application to Loss/PDV Tomography

We now apply our generic results to concrete tomography problems. To apply these results, we need to answer two questions: (i) Does condition (4.11) hold for the problem at hand? (ii) Can we evaluate the coefficient functions in the derived formulas (i.e., $B_C(\boldsymbol{\theta})$, $A_i(\boldsymbol{\theta})$, and $\widetilde{A}_i(\boldsymbol{\theta})$) for a given value of $\boldsymbol{\theta}$? In this subsection, we give positive answers to both questions for loss tomography and PDV tomography.

4.5.4.1 Application to Packet Loss Tomography

Based on the observation model (4.2), we can obtain the per-path FIM $I^{(y)}(\boldsymbol{\theta})$ for loss tomography, whose $(i,\ j)$-th entry equals

$$I_{i,j}^{(y)}(\boldsymbol{\theta}) = \frac{\alpha_y(\boldsymbol{\theta})}{\theta_i\theta_j(1-\alpha_y(\boldsymbol{\theta}))} \mathbb{1}\{i,j \in p_y\}, \tag{4.24}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function. It is easy to verify that this FIM satisfies condition (4.11), and thus the formulas in Theorem 10, Theorem 12, and Corollary 13 apply.

To derive explicit expressions for their coefficients, we take a detailed look at the FIM, which leads to a decomposition into a product of matrices with special structures. Substituting (4.24) into (4.9) gives the (i,j)-th entry of the FIM:

$$I_{i,j}(\boldsymbol{\theta};\boldsymbol{\phi}) = \sum_{y=1}^{|P|} \phi_y \frac{\alpha_y(\boldsymbol{\theta})}{\theta_i\theta_j(1-\alpha_y(\boldsymbol{\theta}))} \mathbb{1}\{i,j \in p_y\}. \tag{4.25}$$

We introduce two auxiliary matrics[9]: $D = \operatorname{diag}\left((d_y)_{y=1}^{|P|}\right)$ for $d_y := \phi_y\alpha_y(\boldsymbol{\theta})/(1-\alpha_y(\boldsymbol{\theta}))$, and $\Theta = \operatorname{diag}(\boldsymbol{\theta})$. Then the above FIM can be written in matrix form as

$$I(\boldsymbol{\theta};\boldsymbol{\phi}) = \Theta^{-1}A^T D A \Theta^{-1}, \tag{4.26}$$

where A is the measurement matrix.

Based on this decomposition, we can evaluate its determinant and trace of the inverse as functions of Θ, A, and D, leading to the following results.

Lemma 14. *Let A_C denote a $|L|\times|L|$ submatrix of the measurement matrix A formed by rows with indices in C ($C \in \mathcal{S}_{|L|}$). Then* $\det(I(\boldsymbol{\theta};\boldsymbol{\phi}))$ *for loss tomography can be expressed as (4.12) with coefficients*

$$B_C(\boldsymbol{\theta}) = \frac{\det(A_C)^2}{\prod_{l\in L}\theta_l^2} \prod_{i\in C} \frac{\alpha_i(\boldsymbol{\theta})}{1-\alpha_i(\boldsymbol{\theta})} \tag{4.27}$$

for each $C \in \mathcal{S}_{|L|}$.

[9]Here $\operatorname{diag}(\mathbf{d})$ denotes a diagonal matrix with the main diagonal $\mathbf{d}$.

Moreover, if $|P| = |L|$ *and* $I(\boldsymbol{\theta}; \boldsymbol{\phi})$ *is invertible, then* $\mathrm{Tr}(I^{-1}(\boldsymbol{\theta}; \boldsymbol{\phi}))$ *can be expressed as (4.17) with coefficients*

$$A_i(\boldsymbol{\theta}) = \frac{1 - \alpha_i(\boldsymbol{\theta})}{\alpha_i(\boldsymbol{\theta})} \sum_{k=1}^{|L|} \theta_k^2 b_{k,i}^2 \tag{4.28}$$

for $i = 1, \ldots, |L|$, *where* $b_{k,i}$ *is the* (k, i)*-th entry of*[10] A^{-1}. *Similarly, the weighted sum of the diagonal elements of* $I^{-1}(\boldsymbol{\theta}; \boldsymbol{\phi})$ *can be expressed as (4.23) with coefficients*

$$\widetilde{A}_i(\boldsymbol{\theta}) = \frac{1 - \alpha_i(\boldsymbol{\theta})}{\alpha_i(\boldsymbol{\theta})} \sum_{k=1}^{|L|} \omega_k \theta_k^2 b_{k,i}^2, \tag{4.29}$$

where ω_k *is the weight of link* l_k.

Proof of Lemma 14. To derive $B_C(\boldsymbol{\theta})$, we evaluate the determinant of the FIM by $\det(\Theta^{-1})^2 \det(A^T D A)$. Applying the Cauchy-Binet formula to $\det(A^T (DA))$ gives

$$\det(I(\boldsymbol{\theta}; \boldsymbol{\phi})) = \frac{1}{\prod_{l \in L} \theta_l^2} \sum_{C \in \mathcal{S}_{|L|}} \det(A_C) \det((DA)_C), \tag{4.30}$$

where similar to A_C, $(DA)_C$ is a $|L| \times |L|$ submatrix of DA formed by rows with indices in C. Since D is diagonal, we can further decompose $\det((DA)_C)$ into $\det(D_C) \det(A_C)$, where $D_C = \mathrm{diag}\,((d_y)_{y \in C})$. Since the only term depending on $\boldsymbol{\phi}$ is $\det(D_C)$, we can rewrite (4.30) as a function of $\boldsymbol{\phi}$ as

$$\det(I(\boldsymbol{\theta}; \boldsymbol{\phi})) = \sum_{C \in \mathcal{S}_{|L|}} \left[\frac{\det(A_C)^2}{\prod_{l \in L} \theta_l^2} \prod_{i \in C} \frac{\alpha_i(\boldsymbol{\theta})}{1 - \alpha_i(\boldsymbol{\theta})} \right] \prod_{i \in C} \phi_i, \tag{4.31}$$

which matches formula (4.12) with $B_C(\boldsymbol{\theta})$ defined as in (4.27).

[10] Given identifiability of $\boldsymbol{\theta}$, A must be invertible in this case; see Section 4.3.2.

To derive $A_i(\boldsymbol{\theta})$, we evaluate the inverse of the FIM by $\Theta A^{-1}D^{-1}A^{-T}\Theta$. Denoting A^{-1} as $(b_{i,j})_{i,j=1}^{|L|}$, we can evaluate the k-th diagonal entry as $I_{k,k}^{-1}(\boldsymbol{\theta};\boldsymbol{\phi}) = \theta_k^2 \sum_{i=1}^{|L|} b_{k,i}^2 d_i^{-1}$ since Θ and D^{-1} are diagonal. Plugging in the definition of d_i^{-1} yields

$$\begin{aligned}\mathrm{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})) &= \sum_{k=1}^{|L|}\theta_k^2 \sum_{i=1}^{|L|} \frac{b_{k,i}^2(1-\alpha_i(\boldsymbol{\theta}))}{\alpha_i(\boldsymbol{\theta})}\cdot\frac{1}{\phi_i} \\ &= \sum_{i=1}^{|L|}\frac{1}{\phi_i}\left[\frac{1-\alpha_i(\boldsymbol{\theta})}{\alpha_i(\boldsymbol{\theta})}\sum_{k=1}^{|L|}\theta_k^2 b_{k,i}^2\right], \end{aligned} \tag{4.32}$$

which matches formula (4.17) with $A_i(\boldsymbol{\theta})$ defined as in (4.28). The same derivation will give the expression for $\widetilde{A}_i(\boldsymbol{\theta})$. □

Remark: For the case of $|P| > |L|$ and invertible $I(\boldsymbol{\theta};\boldsymbol{\phi})$, we can also give an explicit expression for $\mathrm{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi}))$. The key is to plug the decomposed $I(\boldsymbol{\theta};\boldsymbol{\phi})$ into Cramer's formula of calculating $I_{k,k}^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})$ (see (4.19)). Let $A^{(k)}$ denote the submatrix of A by removing the k-th column and $A_C^{(k)}$ the submatrix of $A^{(k)}$ formed by rows with indices in C. A derivation similar to the proof of Lemma 14 shows that

$$\mathrm{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})) = \frac{\sum_{C'\in\mathcal{S}_{|L|-1}}\left[\sum_{k=1}^{|L|}\theta_k^2\det(A_{C'}^{(k)})^2\right]\prod_{i\in C'}d_i}{\sum_{C\in\mathcal{S}_{|L|}}\det(A_C)^2\prod_{i\in C}d_i}, \tag{4.33}$$

which is a rational expression of $\boldsymbol{\phi}$. A similar expression holds for the weighted sum of the diagonal elements of $I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})$.

4.5.4.2 Application to PDV Tomography

Similarly, from the observation model (4.3), we can derive the per-path FIM for PDV tomography as

$$I_{i,j}^{(y)}(\boldsymbol{\theta}) = \frac{1}{2\big(\sum_{l\in p_y}\theta_l\big)^2}\mathbb{1}\{i,j\in p_y\}, \tag{4.34}$$

which also satisfies condition (4.11).

Applying (4.34) to (4.9) gives an expression for individual entries of the FIM for PDV tomography. Observing its similarity to the FIM for loss tomography, we again write it in matrix form by introducing another auxiliary matrix $E = \text{diag}\left((e_y)_{y=1}^{|P|}\right)$ for $e_y := \phi_y/[2(\sum_{l\in p_y}\theta_l)^2]$. It can be verified that the FIM for PDV tomography satisfies $I(\boldsymbol{\theta};\boldsymbol{\phi}) = A^T E A$. This decomposition leads to the following results.

Lemma 15. *The* $\det(I(\boldsymbol{\theta};\boldsymbol{\phi}))$ *for PDV tomography can be expressed as (4.12) with coefficients*

$$B_C(\boldsymbol{\theta}) = \frac{\det(A_C)^2}{2^{|L|}\prod_{i\in C}\left(\sum_{l\in p_i}\theta_l\right)^2} \tag{4.35}$$

for each $C \in \mathcal{S}_{|L|}$ *(A_C defined as in Lemma 14).*

Moreover, if $|P| = |L|$ *and* $I(\boldsymbol{\theta};\boldsymbol{\phi})$ *is invertible, then* $\text{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi}))$ *can be expressed as (4.17) with coefficients*

$$A_i(\boldsymbol{\theta}) = 2\left(\sum_{l\in p_i}\theta_l\right)^2 \sum_{k=1}^{|L|} b_{k,i}^2 \tag{4.36}$$

for $i = 1,\ldots,|L|$ *($b_{k,i}$ is the (k,i)-th entry of A^{-1}). Similarly, the weighted sum of the diagonal elements of* $I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})$ *can be expressed as (4.23) with coefficients*

$$\widetilde{A}_i(\boldsymbol{\theta}) = 2\left(\sum_{l\in p_i}\theta_l\right)^2 \sum_{k=1}^{|L|} \omega_k b_{k,i}^2. \tag{4.37}$$

Proof of Lemma 15. The proof is analogous to that of Lemma 14 by evaluating $\det(A^T E A)$ and $A^{-1}E^{-1}A^{-T}$. □

Remark: Similar to loss tomography, for the case of $|P| > |L|$ and invertible $I(\boldsymbol{\theta};\boldsymbol{\phi})$, we can explicitly write $\text{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi}))$ for PDV tomography as

$$\text{Tr}(I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})) = \frac{\sum_{C'\in\mathcal{S}_{|L|-1}}\left[\sum_{k=1}^{|L|}\det(A_{C'}^{(k)})^2\right]\prod_{i\in C'} e_i}{\sum_{C\in\mathcal{S}_{|L|}}\det(A_C)^2\prod_{i\in C} e_i}, \tag{4.38}$$

which is again a rational expression of $\boldsymbol{\phi}$, and a similar expression holds for the weighted variation.

4.6 Experiment Design Algorithms

The special structures of the design objectives established in Section 4.5 enable us to compute the design parameter $\boldsymbol{\phi}$ efficiently. In the sequel, we will first derive closed-form solutions for the case of $|P| = |L|$, i.e., all probing paths are linearly independent, and then address the case of $|P| > |L|$.

4.6.1 Closed-form Solution for $|P| = |L|$

For the D-optimal design, Theorem 10 implies that when $|P| = |L|$, the determinant of the FIM is proportional to the product of ϕ_i's as shown in (4.16). Since $\sum_{i=1}^{|L|} \phi_i = 1$, by the inequality of arithmetic and geometric means, we see that (4.16) is maximized by setting $\phi_i = 1/|L|$ for all $i = 1, \ldots, |L|$.

Claim 16. *Uniform probing (i.e., $\phi_i = 1/|P|$) is D-optimal when $|P| = |L|$.*

For the A-optimal design, it is easy to show using the Lagrange Multiplier method that a closed-form solution for minimizing (4.17) wrt $\boldsymbol{\phi}$ is the following:

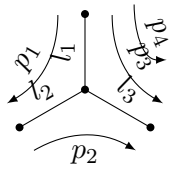
$$\phi_i = \frac{\sqrt{A_i(\boldsymbol{\theta})}}{\sum_{j=1}^{|L|} \sqrt{A_j(\boldsymbol{\theta})}}, \tag{4.39}$$

for $i = 1, 2, \ldots, |L|$. The solution for the weighted A-optimal design is analogous, except that $A_i(\boldsymbol{\theta})$ is replaced by $\widetilde{A}_i(\boldsymbol{\theta})$.

4.6.2 Heuristic Solution for $|P| > |L|$

When $|P| > |L|$, the computation of the optimal design becomes more complicated. From the example in Fig. 4.2, we see that uniform probing is no longer D-optimal. Computing the exact D/A-optimal design involves optimizing a $|P|$-variable

function (4.12) or (4.22), which can only be solved numerically for very small networks. To develop a scalable solution, we leverage the closed-form solution in the case of $|P| = |L|$. We illustrate our idea by a small example in Fig. 4.3. Suppose links l_1, l_2, and l_3 have success rates 0.2, 0.1, and 0.3, respectively. Numerical calculation gives the A-optimal design for inferring these link success rates in the last row of the table. Alternatively, we can select a basis of paths[11] and use the solution in (4.39) to compute the optimal design when only probing paths in the basis; see the first four rows of the table. We see that although the optimal design may use all paths, a design that only optimizes $\boldsymbol{\phi}$ for a properly selected basis can achieve near-optimal performance (see Fig. 4.9 and 4.12 for more comprehensive evaluations).



ϕ_1	ϕ_2	ϕ_3	ϕ_4	$\mathrm{Tr}(I^{-1})$
0.42	0.34	0.24	0	9.70
0.47	0.37	0	0.16	21.79
0.27	0	0.45	0.28	6.95
0	0.22	0.49	0.29	$6.60^{\dagger}$
0.17	0.15	0.44	0.24	5.94^{*}

Figure 4.3: Example for heuristic solution. $*$: A-optimal; †: A-optimal on the best basis.

This observation motivates a two-step heuristic solution, where we first pick a basis of paths that gives the optimal objective value among all bases, and then optimize $\boldsymbol{\phi}$ using solutions in Section 4.6.1 for paths in the basis, while setting $\phi_y = 0$ for paths not in the basis. However, optimizing the basis is itself a combinatorial optimization that is hard to solve exactly. To select a basis, we propose a backward greedy algorithm, given in Algorithm 2. Starting with all $|P|$ paths, it iteratively deselects one path at a time to optimize the design objective (determinant, trace, or weighted trace of $I^{-1}(\boldsymbol{\theta}; \boldsymbol{\phi})$), and the iteration continues until the remaining paths form a basis (lines 2–10). To evaluate the design objective (line 5) before calculating $\boldsymbol{\phi}$, we assume uniform $\boldsymbol{\phi}$ for the selected paths.

[11]Here, 'basis' means a subset of $|L|$ paths that provide an invertible measurement matrix.

Algorithm 2: Two-step Experiment Design for Given $\boldsymbol{\theta}$

1: $P_B \leftarrow P$
2: **for** iteration $i = 1, \ldots, |P| - |L|$ **do**
3: **for** path $p \in P_B$ **do**
4: **if** $P_B \setminus p$ has rank $|L|$ **then**
5: evaluate design objective when only using paths in $P_B \setminus p$
6: record path p^* that yields the optimal objective
7: **end if**
8: **end for**
9: $P_B \leftarrow P_B \setminus p^*$
10: **end for**
11: compute optimal ϕ_y for $p_y \in P_B$; set ϕ_y to 0 for $p_y \notin P_B$

Algorithm 3: Iterative Experiment Design

1: $\phi_y \leftarrow 1/|P|$ for $y = 1, \ldots, |P|$
2: **for** iteration $i = 1, \ldots, N/k$ **do**
3: send k probes according to $\boldsymbol{\phi}$
4: update $\hat{\boldsymbol{\theta}}$ based on probing results
5: compute a new design parameter $\hat{\boldsymbol{\phi}}$ by Algorithm 2 using the updated $\hat{\boldsymbol{\theta}}$
6: update design parameter $\boldsymbol{\phi} \leftarrow (1 - ik/N)\boldsymbol{\phi} + (ik/N)\hat{\boldsymbol{\phi}}$
7: **end for**

4.6.3 Iterative Design Algorithm

In general, the optimal design depends on the unknown parameter $\boldsymbol{\theta}$, which can only be estimated after collecting some measurements. This motivates an iterative design algorithm, presented in Algorithm 3. Specifically, we conduct probing in N/k iterations of k probes each. In each iteration, we send k probes on paths selected according to the current $\boldsymbol{\phi}$ (line 3), update the estimate $\hat{\boldsymbol{\theta}}$ based on the probing results (line 4), and then compute a new design parameter $\hat{\boldsymbol{\phi}}$ using the updated estimate (line 5). During first few iterations, we may not have sufficient measurements to accurately estimate $\boldsymbol{\theta}$, which can mislead our design. To improve robustness against estimation error, we use a combination of the current $\boldsymbol{\phi}$ (obtained from the previous iteration) and the new $\hat{\boldsymbol{\phi}}$ (computed by line 5), and give increasing weight to $\hat{\boldsymbol{\phi}}$ as we obtain more measurements (line 6).

How does the iteratively designed $\hat{\boldsymbol{\phi}}$ converge to the $\boldsymbol{\phi}$ designed based on the true value of $\boldsymbol{\theta}$? Intuitively, as we obtain more measurements, the estimated $\hat{\boldsymbol{\theta}}$ will converge to $\boldsymbol{\theta}$, and thus the iteratively designed $\hat{\boldsymbol{\phi}}$ will converge to the $\boldsymbol{\phi}$ optimized for $\boldsymbol{\theta}$. Formalizing this intuition requires two steps: first, we need to show that the design objectives (e.g., trace of the inverse FIM) computed from $\hat{\boldsymbol{\theta}}$ and $\boldsymbol{\theta}$ will converge so that we will select the correct basis P_B; moreover, we need to show that for a fixed P_B, the optimal ϕ_y ($p_y \in P_B$) based on $\hat{\boldsymbol{\theta}}$ and $\boldsymbol{\theta}$ will converge. We now provide concrete analysis for loss and PDV tomography. We only consider the A-optimal design due to space limitation, as results are analogous for the other design objectives.

Theorem 17. *For both loss and PDV tomography, as the number of probes per path increases, the estimated objective of the A-optimal design (i.e., trace of the inverse FIM based on $\hat{\boldsymbol{\theta}}$) converges to the true objective with high probability. Moreover, for a fixed basis P_B, the A-optimal design on P_B based on $\hat{\boldsymbol{\theta}}$ converges to the true A-optimal design on P_B based on $\boldsymbol{\theta}$ with high probability.*

Proof of Theorem 17 . Fundamental to our proof is the convergence of empirical path parameters to the true parameters. For loss tomography, these are path success rates, denoted by $\boldsymbol{\alpha}$; for PDV tomography, these are path PDV variances, denoted by $\boldsymbol{\sigma}$. Based on the *Chernoff-Hoeffding bound*, the empirical parameters converge exponentially fast as the number of probes n_i for each path p_i ($i = 1, \ldots, |P|$) increases, i.e., both $\Pr\{|\hat{\alpha}_i - \alpha_i| \leq \delta\}$ and $\Pr\{|\hat{\sigma}_i - \sigma_i| \leq \delta\}$ are lower bounded by $1 - 2e^{-2\delta^2 n_i}$ ($i = 1, \ldots, |P|$). What remains is to bound the error in the design objective and $\boldsymbol{\phi}$, given δ-error in estimating α_i and σ_i. Due to space limitation, we only detail the analysis for loss tomography, as the analysis for PDV tomography is analogous but simpler.

Let $T(\boldsymbol{\theta})$ denote the trace of inverse FIM based on uniform $\boldsymbol{\phi}$ (as assumed in line 5 of Algorithm 2). For a function $x(\boldsymbol{\theta})$, we use $\hat{x}$ to denote $x(\hat{\boldsymbol{\theta}})$. We will show that for

any sufficiently small $\delta > 0$, $\exists \epsilon_1(\delta)$, $\epsilon_2(\delta)$ that go to 0 as $\delta \to 0$ such that $|\hat{\alpha}_i - \alpha_i| \leq \delta$ $(i = 1, \ldots, |P|)$ implies $|\hat{T} - T| \leq \epsilon_1(\delta)$, and $|\hat{\phi}_i - \phi_i| \leq \epsilon_2(\delta)$ for all $p_i \in P_B$.

For loss tomography, a derivation similar to Lemma 14 shows that

$$T = \frac{|P| \sum_{C' \in \mathcal{S}_{|L|-1}} \Big[\sum_{k=1}^{|L|} \theta_k^2 \det(A_{C'}^{(k)})^2 \Big] \prod_{i \in C'} \frac{\alpha_i}{1-\alpha_i}}{\sum_{C \in \mathcal{S}_{|L|}} \det(A_C)^2 \prod_{i \in C} \frac{\alpha_i}{1-\alpha_i}}, \tag{4.40}$$

where $A^{(k)}$ denotes the submatrix of A by removing the k-th column and $A_C^{(k)}$ the submatrix of $A^{(k)}$ formed by rows with indices in C. Denoting the numerator of (4.40) by f_1 and the denominator by f_2 (both functions of $\boldsymbol{\theta}$). It can be shown that δ-error in α_i implies $|\hat{\theta}_k - \theta_k| \leq e^{\delta'} - 1 := c_0(\delta)$ $(k = 1, \ldots, |L|)$, where δ' is the largest absolute value for entries of $(A^T A)^{-1} A^T \frac{\delta}{\boldsymbol{\alpha} - \delta}$ ($\frac{\delta}{\boldsymbol{\alpha} - \delta}$ is a column vector defined as $(\frac{\delta}{\alpha_i - \delta})_{i=1}^{|P|}$). Moreover, $|\prod_{i \in C} \frac{\hat{\alpha}_i}{1-\hat{\alpha}_i} - \prod_{i \in C} \frac{\alpha_i}{1-\alpha_i}| \leq \max(\prod_i \alpha_i - \prod_i (\alpha_i - \delta), \prod_i (\alpha_i + \delta) - \prod_i \alpha_i) / \prod_i (1 - \alpha_i - \delta)(1 - \alpha_i) := c_1(\delta; C)$. Based on these results, we have

$$\begin{aligned} |f_1 - \hat{f}_1| \leq & |P| \sum_{C' \in \mathcal{S}_{|L|-1}} \Big[\sum_{k=1}^{|L|} \theta_k^2 \det(A_{C'}^{(k)})^2 c_1(\delta; C') \\ & + 2c_0(\delta) \Big(\prod_{i \in C'} \frac{\alpha_i}{1-\alpha_i} + c_1(\delta; C') \Big) \sum_{k=1}^{|L|} \det(A_{C'}^{(k)})^2 \Big] := c_2(\delta), \end{aligned} \tag{4.41}$$

and $|f_2 - \hat{f}_2| \leq \sum_{C \in \mathcal{S}_{|L|}} \det(A_C)^2 c_1(\delta; C) := c_3(\delta)$. Together, these bounds yield

$$|T - \hat{T}| \leq \frac{f_1 c_3(\delta) + f_2 c_2(\delta)}{f_2 (f_2 - c_3(\delta))} := \epsilon_1(\delta), \tag{4.42}$$

which goes to 0 as $\delta \to 0$ since $c_i(\delta) \to 0$ $(i = 0, \ldots, 3)$.

Given a basis P_B, the A-optimal designs on P_B, calculated by (4.39), based on $\hat{\boldsymbol{\theta}}$ and $\boldsymbol{\theta}$ satisfy

$$|\phi_i - \hat{\phi}_i| \leq \frac{\epsilon \Big(|L| \sqrt{A_i(\boldsymbol{\theta})} + \sum_{j=1}^{|L|} \sqrt{A_j(\boldsymbol{\theta})} \Big)}{\Big(\sum_{j=1}^{|L|} \sqrt{A_j(\boldsymbol{\theta})} \Big) \Big(\sum_{j=1}^{|L|} \sqrt{A_j(\boldsymbol{\theta})} - \epsilon |L| \Big)} \tag{4.43}$$

if $|\sqrt{A_i(\boldsymbol{\theta})} - \sqrt{A_i(\hat{\boldsymbol{\theta}})}| \leq \epsilon$ for all $p_i \in P_B$ for a sufficiently small $\epsilon > 0$. Based on the expression of $A_i(\boldsymbol{\theta})$ in Lemma 14, we can show that $|\hat{\alpha}_i - \alpha_i| \leq \delta$ implies

$$|A_i(\boldsymbol{\theta}) - A_i(\hat{\boldsymbol{\theta}})| \leq \frac{2(1-\alpha_i)\beta_i c_0(\delta)}{\alpha_i} + \frac{\beta_i \delta}{\alpha_i(\alpha_i - \delta)}, \tag{4.44}$$

where $\beta_i := \sum_k b_{k,i}^2$. Hence, $|\sqrt{A_i(\boldsymbol{\theta})} - \sqrt{A_i(\hat{\boldsymbol{\theta}})}| \leq \epsilon(\delta)$ for $\epsilon(\delta) := \max_i \frac{2(1-\alpha_i)\beta_i c_0(\delta)}{\alpha_i \sqrt{A_i(\boldsymbol{\theta})}} + \frac{\beta_i \delta}{\alpha_i(\alpha_i - \delta)\sqrt{A_i(\boldsymbol{\theta})}}$. Plugging $\epsilon(\delta)$ into (4.43) gives a bound on $|\phi_i - \hat{\phi}_i|$, denoted by $\epsilon_2(\delta)$, that goes to 0 as $\delta \to 0$. □

4.7 Performance Evaluation

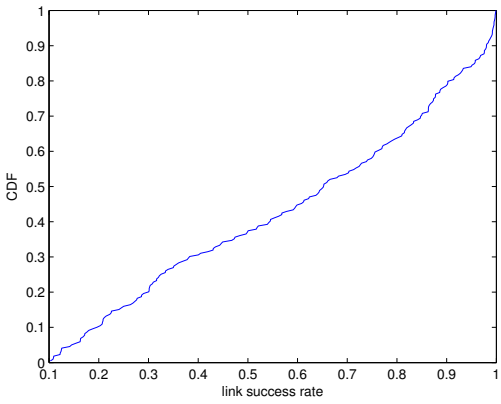
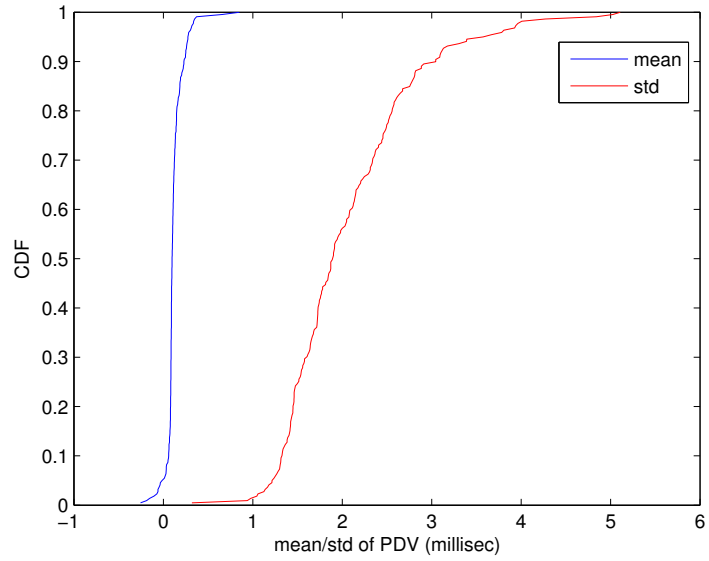


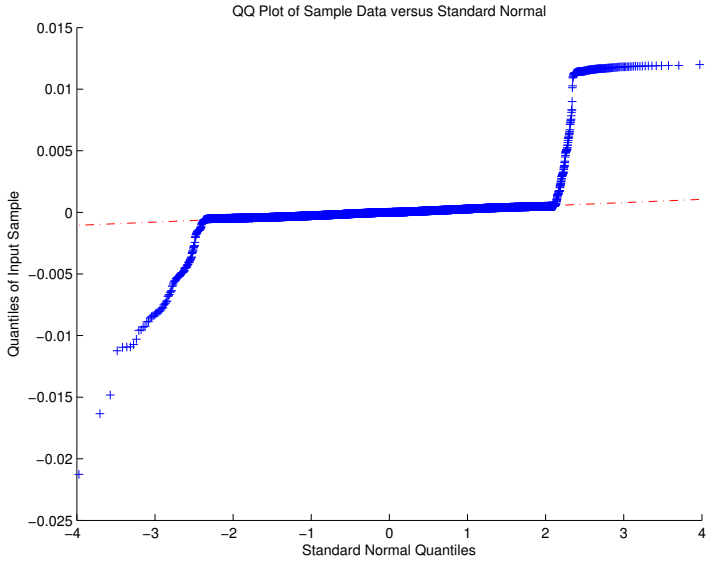
Figure 4.4: Distribution of Roofnet link success rates.

We evaluate different experiment designs by packet-level simulations on real network topologies and link parameters. Our goal in the evaluation is two-fold: (i) evaluating the performance of (iterative) A-optimal design compared with uniformly allocating probes (*uniform probing*), and (ii) evaluating the impact of system parameters such as link weights, number of monitors, and number of paths.

To guarantee identifiability of all the link parameters, we first place a minimum set of monitors by the *Minimum Monitor Placement (MMP)* algorithm in [33] and then place the remaining monitors, if any, randomly. Given the monitors, we first construct



(a) mean/std of link PDV (ms)



(b) Q-Q plot

Figure 4.5: Distribution of Roofnet link PDVs.

$|L|$ linearly independent paths by the *Spanning Tree-based Path Construction (STPC)* algorithm in [34] and then construct additional paths if needed by a random walk[12].

[12]We remove cycles from the random-walk paths so that all paths are cycle-free, although this is not required for the design of probe allocation.

We consider two types of link weights: *homogeneous* link weights, where all links have unit weight, and *heterogeneous* link weights, where a randomly selected subset of K links have a larger weight Ω ($\Omega > 1$), and the rest of the links have unit weight. In the case of heterogeneous link weights, we set $K = 1$ and $\Omega = 500$.

We measure the performance of an experiment design by the (weighted) average MSE and bias over all estimated link parameters when applying the MLE (Section 4.4) to measurements collected using this design. Furthermore, we evaluate the CRB and the design parameter $\boldsymbol{\phi}$ to gain insights on the internal behaviors of various designs. All results are averaged over 5 instances of monitor locations, measurement paths, and link weights, and 100 Monte Carlo runs of probing simulations per instance. In each Monte Carlo run, we simulate 10^5 probes, which is divided into 100 iterations of 1000 probes each for the iterative design.

4.7.1 Dataset for Evaluation

To evaluate our experiment design in a realistic scenario, we use the Roofnet dataset [2], which contains topologies and link measurements from a 38-node wireless mesh network. The dataset contains four subsets of data, corresponding to data rates 1, 2, 5.5, and 11 Mbps. We only present results based on the 1-Mbps data, as the results are similar to those for the other data rates. The raw dataset contains sent/received packet sequence numbers and timestamps between all pairs of nodes within communication range.

This dataset is suitable for evaluating both loss tomography and PDV tomography. For loss tomography, we extract link success rates by computing the fraction of packets sent by a first node that are received by a second node. For PDV tomography, we extract link PDVs by computing the difference between inter-packet delays at a sender and a receiver (ignoring lost packets). We then take the average of both directions

of transmission as the parameter of a link[13]. We also filter out links with success rates below 0.1 to only focus on useful links. After filtering, we obtain a topology with 38 nodes and 219 (undirected) links; see Fig. 4.4 and 4.5 (a) for distributions of the link parameters. We also compare the empirical PDV distribution per link with the normal distribution; see Fig. 4.5 (b) for the Quantile-Quantile (Q-Q) plot for a sample link (dashed line corresponds to a true normal distribution). We see that the mean PDVs are much smaller than the std's, and that the majority (90%+) of the PDV values fit a normal distribution (similar results are observed for other links), both confirming our zero-mean normal assumption in Section 4.2.2.2.

4.7.2 Evaluation of Loss Tomography

We first evaluate the performance of different designs as the number of probes increases; see Fig. 4.6 for results under homogeneous link weights, and Fig. 4.7 for results under heterogeneous link weights. From Fig. 4.6 (a) and 4.7 (a), we see that the A-optimal design and its iterative version achieve lower MSE than uniform probing, and the improvement is greater under heterogeneous link weights. Examining the design parameter $\boldsymbol{\phi}$ under each design (Fig. 4.6 (c) and 4.7 (c)) verifies that this improvement is achieved through nonuniformly allocating probes to better measure the paths that provide more information for estimating link success rates (paths are sorted in the order of increasing probing probabilities under the A-optimal design). The same figure also verifies that the iterative design is able to converge to the true A-optimal design (their curves essentially overlap); we will evaluate the rate of convergence later. Interestingly, for loss tomography, the MLE (Eq. (5.2)) is biased at finite sample sizes as shown in Fig. 4.6 (b) and 4.7 (b), and thus the CRB does not provide a true lower bound on the MSE as shown in Fig. 4.6 (a) and 4.7 (a).

[13]Note that the realizations of link losses/PDVs for each probe are generated according to our model, using parameters extracted from the dataset.

Nevertheless, the CRB captures trends of the MSE so that minimizing the CRB provides a design (i.e., A-optimal design) with low MSE.

Table 4.1: Relative Performance for Loss Tomography (20 monitors, 10^5 probes)

Link weights	$\frac{\text{CRB}^A}{\text{CRB}^U}$	$\frac{\text{MSE}^A}{\text{MSE}^U}$	$\frac{\text{MSE}^I}{\text{MSE}^U}$
homogeneous	0.12	0.55	0.58
heterogeneous	0.18	0.41	0.35

To better appreciate the advantage of A-optimal design, we summarize the relative performance of the A-optimal and the iterative A-optimal designs compared with uniform probing, measured by ratios of their CRB and MSE (the lower, the better); see Table 4.1, where $\{\cdot\}^A$ stands for A-optimal, $\{\cdot\}^U$ for uniform, and $\{\cdot\}^I$ for iterative A-optimal. We see that although the CRB overestimates performance improvement, our iterative design algorithm used in conjunction with the A-optimal criterion indeed achieves a much lower MSE than uniform probing (40–65% lower). Since our design takes into account different link weights, it achieves greater improvement in the case of heterogeneous link weights.

Next, we study the impact of system parameters on estimation performance. We first vary the number of monitors and repeat the probing simulation under each instance of monitor placement. Fig. 4.8 (a)–(b) show the error bar plots of MSE/CRB and absolute bias computed over different instances of monitor placement and path construction. The result shows that all probing methods benefit as we place more monitors. Intuitively, this is because with more monitors, paths become shorter, making the measurements less aggregated and more informative for inferring parameters of individual links. We also evaluate the impact on convergence rate of the iterative design by measuring the L_2-distance of the iterative design ($\boldsymbol{\phi}^I$) to the A-optimal design ($\boldsymbol{\phi}^A$) across iterations; see Fig. 4.8 (c). The result verifies that the iterative design algorithm is able to quickly converge to the true A-optimal design. Moreover,

the convergence becomes faster as the number of monitors increases, because a larger number of monitors allows more accurate estimation of the link parameters and thus closer approximation of the true A-optimal design. We only show the results under homogeneous link weights, as the observations are analogous under heterogeneous link weights.

So far we have limited probing to a basis of paths. To evaluate the impact of probing extra paths, we add paths constructed by a random walk (#extra paths $= |P| - |L|$) and repeat the simulations; see Fig. 4.9. Since when $|P| > |L|$, the A-optimal design can no longer be computed in closed form, we only compute a constrained A-optimal design on a basis selected by Algorithm 2 (based on the true value of $\boldsymbol{\theta}$), simply referred to as 'A-optimal'. Due to the higher complexity of Algorithm 2 in this case, we reduce the number of iterations to 20, each with 5000 probes. We see that although the constrained A-optimal design given by Algorithm 2 only probes a subset of paths (i.e., a basis), it still performs notably better than uniform probing which probes all the paths by strategically allocating probes (Fig. 4.9 (a)–(b)). In contrast to adding monitors, adding paths does not significantly impact the MSE; we do not observe a clear trend in bias. Meanwhile, we see from Fig. 4.9 (c) that having extra paths significantly slows down convergence of the iterative design. Detailed examination shows that this is due to near-tie between some bases in terms of the objective value (trace of the inverse FIM). Nevertheless, Fig. 4.9 (a) shows that the iterative design outperforms uniform probing in terms of MSE. We have obtained similar results under heterogeneous link weights (omitted due to space limitation).

4.7.3 Evaluation of PDV Tomography

We have evaluated PDV tomography in a similar manner. Specifically, Fig. 4.10 shows the performance wrt number of probes in a basic setting with homogeneous link weights, Fig. 4.11 shows the impact of placing more monitors, and Fig. 4.12

Table 4.2: Relative Performance for PDV Tomography (20 monitors, 10^5 probes)

Link weights	$\frac{\text{CRB}^A}{\text{CRB}^U}$	$\frac{\text{MSE}^A}{\text{MSE}^U}$	$\frac{\text{MSE}^I}{\text{MSE}^U}$
homogeneous	0.47	0.47	0.48
heterogeneous	0.38	0.39	0.39

shows the impact of probing more paths. We have obtained similar results under heterogeneous link weights (omitted). Overall, the relative performances of different designs are similar to those for loss tomography, but the absolute performances differ.

Specifically, the estimation error for PDV tomography decays faster than that for loss tomography as the number of probes increases, as each measurement (path PDV) contains more fine-grained information about the links. As a consequence, the MSE values in Fig. 4.10 (a) are much smaller than those in Fig. 4.6 (a) for the same number of probes. A more striking difference between the two plots is that instead of being a loose approximation of MSE as in loss tomography, the CRB accurately predicts the MSE in PDV tomography (the curves overlap). This is because the estimator for PDV tomography (Eq. (4.8)) is unbiased, as verified by Fig. 4.10 (b); note that the empirical bias is negligible compared to the parameters of interest (link PDV variances). As in loss tomography, the A-optimal design for PDV tomography leads to a highly skewed distribution of probes across paths, as shown in Fig. 4.10 (c). Similar to Table 4.1 for loss tomography, we summarize the relative performance for PDV tomography in Table 4.2, which shows that the iterative design achieves a similar improvement of 50–60% for PDV tomography, but the performance predicted by the CRB is much more accurate.

As we vary the number of monitors, we again see a clear trend of decreasing CRB/MSE in Fig. 4.11 (a). A key difference from the results for loss tomography (Fig. 4.8 (a)) is that the CRB accurately predicts the value of the MSE. A more subtle difference is that as we increase the number of monitors, the gap between (iterative)

A-optimal and uniform probing becomes narrower, instead of becoming wider as in loss tomography. We also see a trend of slightly decreasing absolute bias, and that the iterative design incurs a slightly larger bias; see Fig. 4.11 (b). Note, however, that the difference in bias is insignificant as the estimator is statistically unbiased. Another difference from loss tomography is that the convergence rate of iterative design for PDV tomography is largely independent of the number of monitors, as shown in Fig. 4.11 (c).

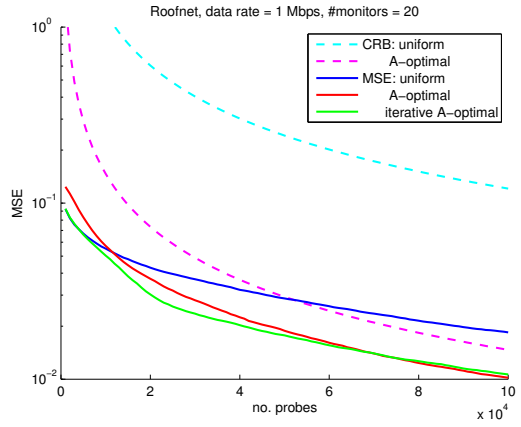
As we vary the number of paths, we see from Fig. 4.12 (a) that the MSE of uniform and A-optimal probing (on a basis selected by Algorithm 2) remains largely the same, so does their CRB. We notice a mild but notable increase in the MSE of the iterative A-optimal design, because having extra paths slows down the convergence of the design parameter, as shown in Fig. 4.12 (c). Note that the convergence is much faster than that in loss tomography (Fig. 4.9 (c)), because the parameters of interest (link PDV variances) can be estimated more accurately using the same number of probes (see Fig. 4.10 (a) and 4.6 (a)). As in loss tomography, increasing the number of paths does not have monotone impact on the bias as shown in Fig. 4.12 (b).

4.8 Conclusion

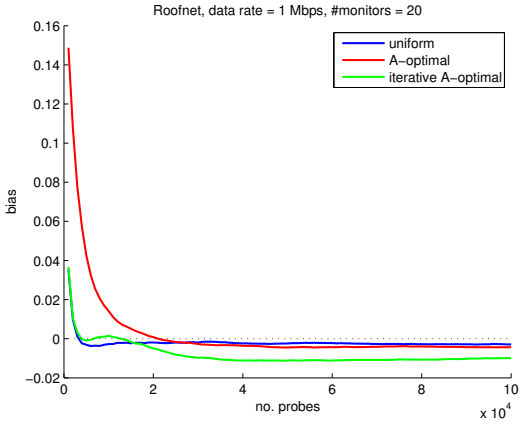
We propose a general framework of optimal experiment design for inferring parameters of stochastic link metrics using path measurements, with two concrete case studies on loss tomography and PDV tomography. Using the FIM to measure the amount of information contained in each measurement, we formulate the problem as an optimization of probe distribution across paths, with two widely-adopted objectives known as D-optimality and A-optimality. We are particularly interested in A-optimal design since it is directly linked to MSE and can be easily extended to incorporate different link weights. Under certain conditions on the FIM, satisfied for both loss and PDV tomography, we derive explicit expressions for both objectives

as functions of the design parameter, which enable closed-form solution of the optimal design when the probing paths are linearly independent. Using this solution as a building block, we develop a two-step heuristic and an iterative algorithm to address the issues of linearly dependent paths and dependency on unknown parameters. Our evaluations on real datasets verify the effectiveness of the proposed solution in reducing MSE, even if the FIM-based bound can be loose.

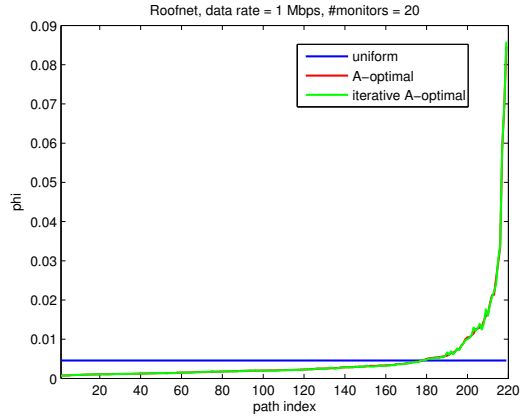
Discussion: While our design is based on probabilistic allocation of probes, our solution can be easily modified for deterministic probe allocation. Specifically, our formulas for the design objectives derived in Section 4.5 remain valid when replacing the probing probability ϕ_y by the allocated number of probes N_y for each path p_y. Based on these formulas, one can derive analogous solutions to $(N_y)_{y=1}^{|P|}$, under the new constraints that $\sum_{y=1}^{|P|} N_y = N$ (N: total number of probes) and N_y's are integers. Relaxing the integer constraint yields $N_y = \phi_y N$, where ϕ_y is the design parameter computed by our current solution, rounding of which leads to a deterministic probe allocation. However, deterministic probe allocation faces an additional challenge in iterative design, where the *order* of probing also needs to be optimized to obtain useful estimates as early as possible. In this regard, the probabilistic design framework simplifies the design process.



(a) MSE and CRB

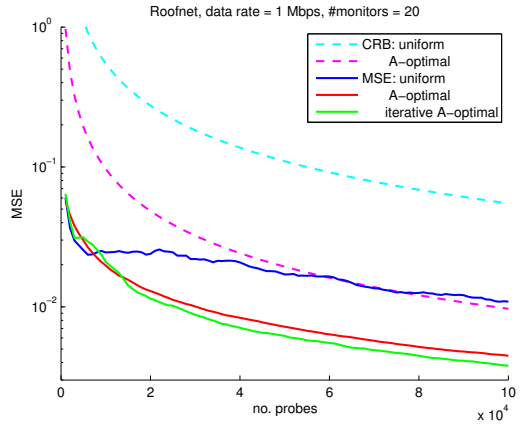


(b) Bias

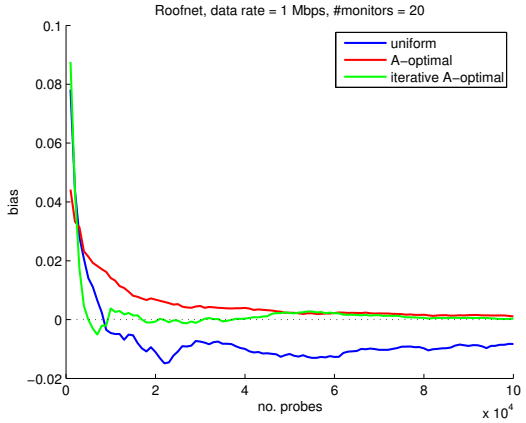


(c) ϕ

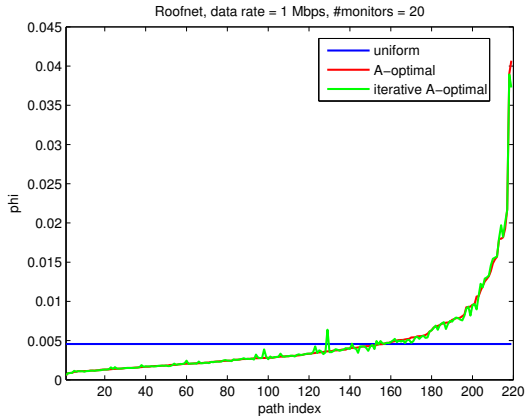
Figure 4.6: Loss tomography, homogeneous link weights (20 monitors, 219 paths)



(a) MSE and CRB

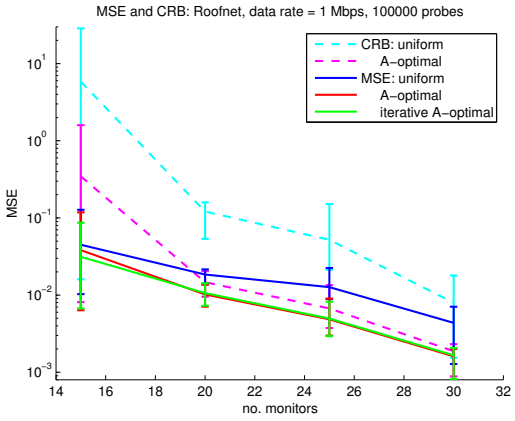


(b) Bias

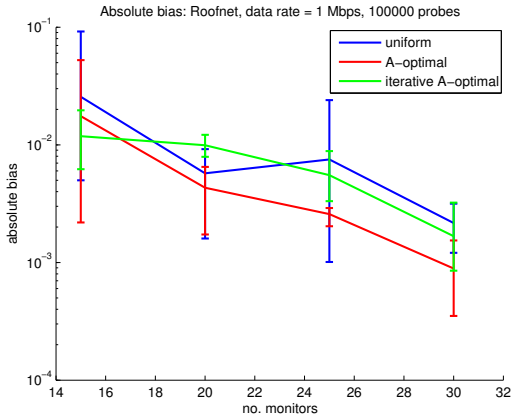


(c) ϕ

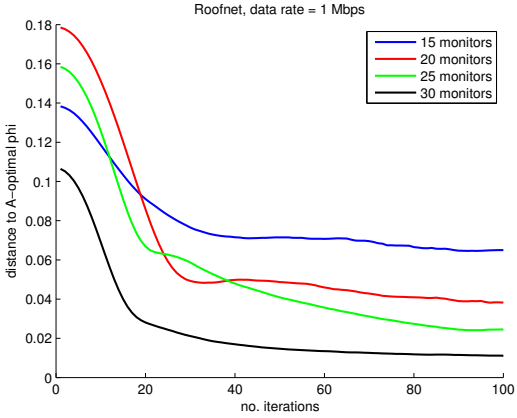
Figure 4.7: Loss tomography, heterogeneous link weights (20 monitors, 219 paths)



(a) MSE and CRB

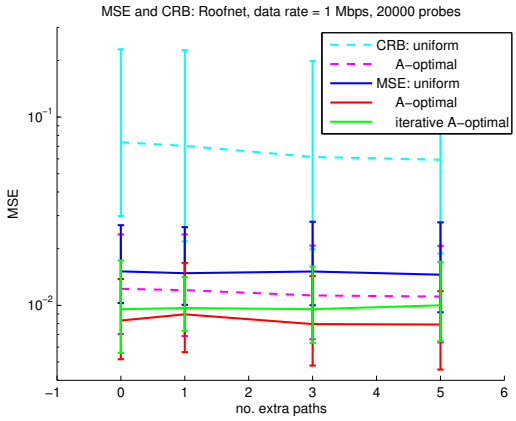


(b) Absolute bias

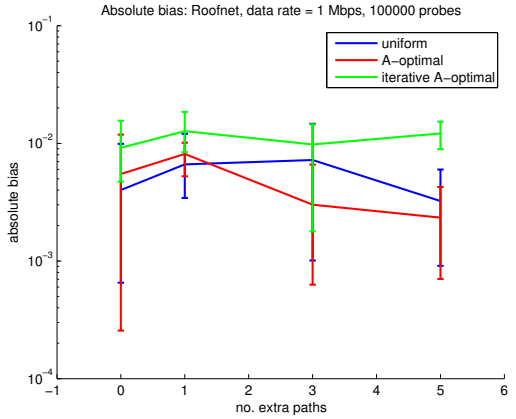


(c) Convergence of $\boldsymbol{\phi}^I$

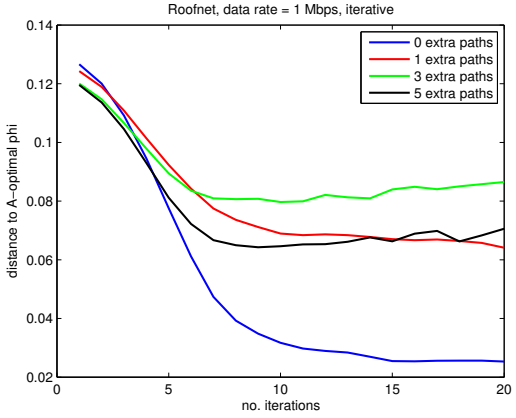
Figure 4.8: Loss tomography, varying number of monitors (219 paths, 10^5 probes, homogeneous link weights)



(a) MSE and CRB

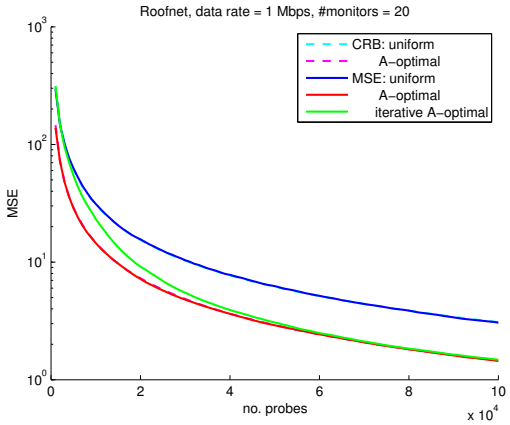


(b) Absolute bias

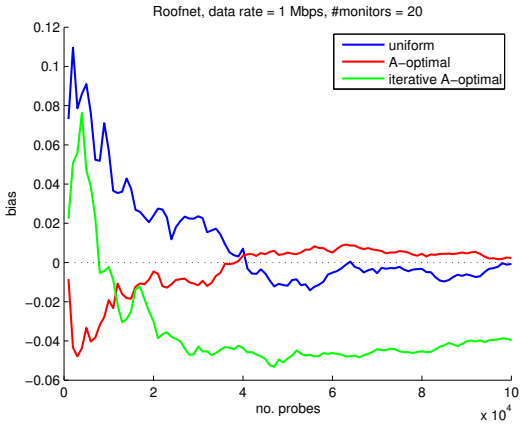


(c) Convergence of ϕ^I

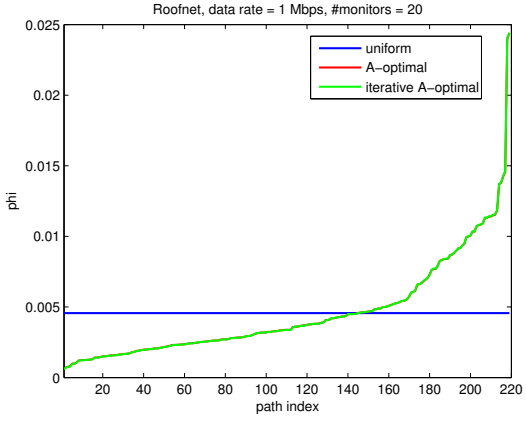
Figure 4.9: Loss tomography, varying number of paths (20 monitors, 10^5 probes, homogeneous link weights)



(a) MSE and CRB

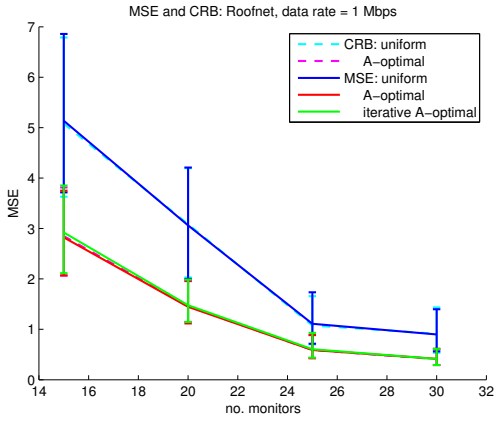


(b) Bias (mean $\theta_l = 4.8$)

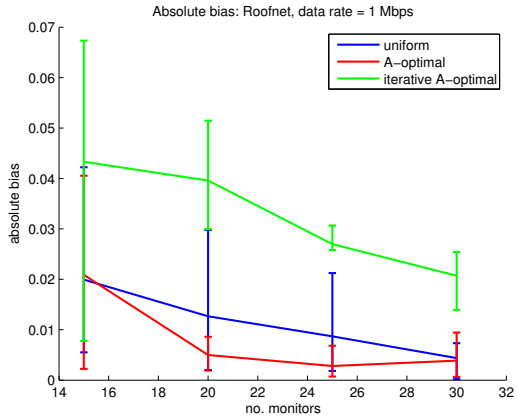


(c) ϕ

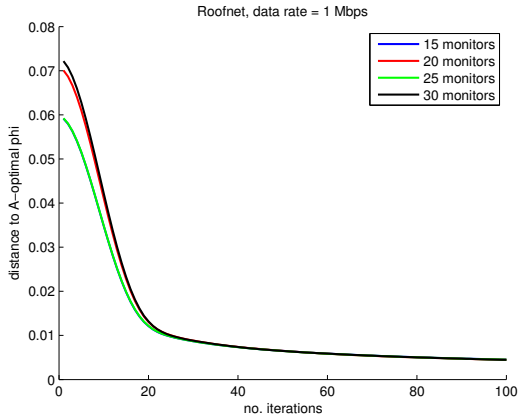
Figure 4.10: PDV tomography, varying number of probes (20 monitors, 219 paths, homogeneous link weights)



(a) MSE and CRB

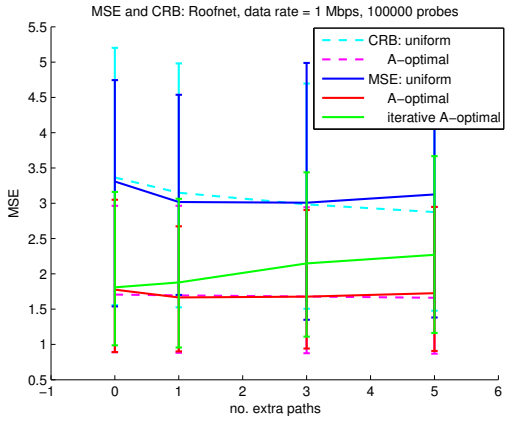


(b) Absolute bias

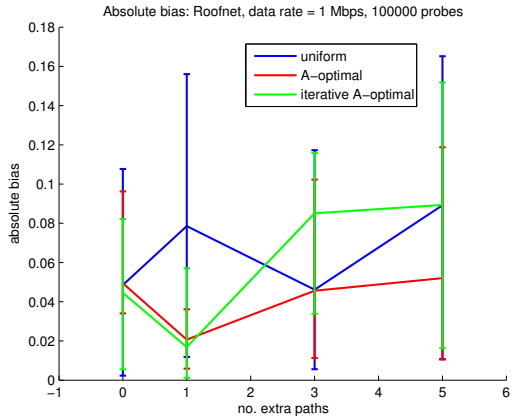


(c) Convergence of $\boldsymbol{\phi}^I$

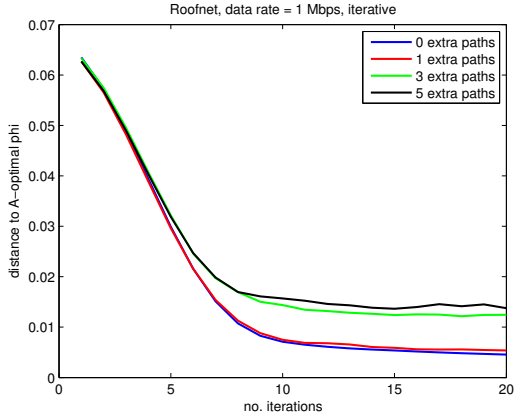
Figure 4.11: PDV tomography, varying number of monitors (219 paths, 10^5 probes, homogeneous link weights)



(a) MSE and CRB



(b) Absolute bias



(c) Convergence of $\boldsymbol{\phi}^I$

Figure 4.12: PDV tomography, varying number of paths (20 monitors, 10^5 probes, homogeneous link weights)

CHAPTER 5

MULTICAST VS. UNICAST FOR LINK LOSS TOMOGRAPHY

5.1 Introduction

We introduced network tomography and the inference it addresses in the previous chapter. The most adopted and investigated measurement methods for network tomography are unicast measurement [46] and multicast measurement [10, 9, 18, 32]. Unicast tomography gathers independent measurements on multiple end-to-end paths via unicast probes and inverts path performance metrics to estimate corresponding link performance metrics. Multicast tomography, on the other hand, gathers correlated measurements along multicast trees between each source and its corresponding receivers via multicast probes. In networks that do not directly support multicast communications, multicast-like measurements can be obtained by sending batches of back-to-back unicast probes, referred to as correlated unicast, so that probes in the same batch experience similar performance on the same link [17]. For each of the above probing methods, there have been studies on how to allocate probes across different paths/trees so that the overall information about the link parameters of interest can be maximized [23, 51, 25]. There is, however, a lack of understanding of when each of these methods is preferable to the other.

Intuitively, multicast is always preferable to correlated unicast as it generates less traffic in obtaining the same measurements. Comparison between multicast (or correlated unicast) and unicast is much less straightforward: on one hand, multicast can provide more end-to-end measurements than unicast for the same amount of

probing traffic (measured by the total number of hops traversed by probes); on the other hand, unicast provides more fine-grained control over the distribution of probes (at the level of paths rather than trees), which allows one to focus probing resources on paths providing more information about link parameters or containing links of higher importance.

This chapter of the thesis aims to provide an initial understanding of the strengths and weaknesses of each probing method for inferring link loss rates in networks with tree topologies. The tree topology represents a case of special interest in network tomography. Besides its simplicity, the tree topology is shown to approximate latency and bandwidth in the Internet [43], and most tomography-based topology discovery methods generate logical topologies that are trees [28]. Given a network spanned by a single multicast tree, we ask the following questions: (i) Can unicast probing consistently estimate link loss rates? (ii) If so, how should we allocate the unicast probes among different paths? (iii) How do different probing methods compare in terms of the accuracy of estimated link loss rates, and how does the comparison depend on parameters such as the probing budget, the network size, and the values of link loss rates?

5.1.1 Related Work

In existing works, statistical tomography models each link metric as a random variable with a (partially) unknown probability distribution, and applies various estimation techniques to infer the distribution from path measurements. When supported, multicast-based probing has been proposed to estimate link parameters from measurements at multicast receivers [10, 9, 18, 32]. Specifically, [10] derives a maximum likelihood estimator to infer link loss rates from packet losses observed at receivers of a single-source multicast tree, which is later extended to use losses observed from multiple trees in [9]. Analogous results have been obtained for delays, where packet

delays observed at multicast receivers are used to infer variances or distributions of delays at internal links [18, 32]. Multicast probing has the benefit of uniquely identifying metrics of link segments between branching points [9], but it also has the limitation of requiring network layer support for multicast communications. To relax this limitation, [17] has proposed a technique to emulate multicast using back-to-back unicast probes referred to as *correlated unicast* in this chapter, under the assumption that unicast probes sent sufficiently close to each other (in time) on a given path will experience the same realization of losses on each link of the path.

Both multicast probing and correlated unicast probing have the drawback that they require sophisticated coordination at the network layer. In contrast, unicast probing only measures the pairwise performance between individual source-destination pairs and is generally supported by any network. Under the assumption that probed paths form a full-rank measurement matrix, [46] has shown that it is possible to infer link delay distributions solely from end-to-end delays of (independent) unicast probes. Recently, there have been tremendous advances in techniques to ensure the full-rank assumption, including techniques to transform the original topology into a logical topology such that the measurement matrix on the logical topology has full rank [53], and techniques to construct unicast paths (under the assumption of controllable routing) such that the measurement matrix has full rank [34].

The theory of experiment design for general statistical inference casts the problem as an optimization of a set of design objectives that capture various aspects of estimation accuracy [4]. The approach has recently been applied to design experiments for network tomography. Under multicast or correlated unicast, [23, 51] have proposed to measure the quality of an experiment design by appropriate functions of the Fisher Information Matrix (FIM) and to design probing experiments such that certain performance criteria based on the FIM can be optimized (A-optimality in [23], D-optimality in [51]). These solutions either rely on numerical solvers [51] or a coarse

approximation that ignores off-diagonal elements of the FIM [23]. This approach has recently been extended to unicast probing, where closed-form solutions are derived to optimally allocate probes among unicast paths under the criterion of D-optimality or A-optimality [25].

We note that we focus on inferring link parameters (loss rates) from multicast or unicast probes with known topology. There is another line of work on inferring (routing) topology of a network from end-to-end observations, where most work assumes multicast or emulated multicast (by back-to-back unicast) probes. See [28] and references therein for more details.

5.1.2 Summary of Contributions

In investigating multicast and unicast on tree structures and comparing their performance, our specific contributions are:

1. We establish the identifiability of all links in trees without degree-2 nodes using unicast probes between leaves, and propose a path construction algorithm to achieve identifiability.

2. We derive a closed-form expression for optimal probe allocation for unicast probing.

3. We derive explicit formulas for evaluating the FIM for both unicast and multicast probing.

4. We use packet-level simulation to evaluate the performance of unicast, multicast and correlated unicast under varying system parameters including link weights, link success rates and tree size. Besides confirming that multicast always outperforms correlated unicast, our results show that unicast outperforms multicast when the probing budget measured in total number of hops traversed by probes is small. This is especially true when links have heterogeneous weights. On the

other hand, multicast and correlated unicast are more robust than unicast to different link success rates and different tree sizes.

5.2 Loss Tomography on Trees

5.2.1 Network Model

igh-level question: What is the performance, measured by the *Cramér-Rao bound (CRB)* and MSE, of multicast probing, emulated by back-to-back unicast probing along multicast trees, compared with unicast probing? The answer is non-trivial because on one hand, emulated multicast probing can identify a larger number of links by leveraging correlation between concurrent probes [10]; on the other hand, unicast probing gives more flexibility to design experiments and optimize allocation of probes. In this comparison, we only consider loss tomography. Recall that θ_l denotes the success rate of link l, and $\alpha_i := \prod_{l \in p_i} \theta_l$ the success rate of path p_i.

Let $\mathcal{T} = (V, L)$ denote a directed tree with nodes V and links L. Let $s \in V$ denote the source which is the root of the tree and $R \subset V$ the receivers which are leaf nodes. The complement of the leaf nodes $I = V - R$ is referred to as the internal nodes. Following [10], we assume that the other nodes (referred to as *branching nodes*) have degrees of at least three. Without loss of generality, we label the nodes so that $s = 0$ and $R = \{|L| - |R| + 1, \ldots, |L|\}$. We label the links so that link i is the link leading to node i (from node 0). We refer to node 0 as the *root* of the tree and R as the set of *leaves*. Let $f(i)$ denote the parent of node i in the tree, $d(i)$ the set of children of node i, and $s(i)$ the set of siblings of node i (i.e., $s(i) = \{j \in d(f(i)) : j \neq i\}$). Losses on link $l \in L$ follow a Bernoulli process with an (unknown) loss probability $1 - \theta_l$, θ_l being the success probability. We assume that losses are independent from link to link, and across time. We also assume symmetric loss rates on both directions for each link.

5.2.2 Observation model and MLE of unicast

Let P denote the set of paths on which the monitoring system can inject probes on and observe the end-to-end performance. Link success rates are then inferred from unicast measurements on paths. Following the definition in [25], the *measurement matrix* is a $|P| \times |L|$ matrix $A := [A_{y,l}]$, defined by P, where $A_{y,l} = 1$ if link l is on path y and $A_{y,l} = 0$ otherwise. We use the same probabilistic design model as in [25], where each probe is sent over a path y randomly selected from P with probability ϕ_y. Here $\boldsymbol{\phi} := (\phi_y)_{y=1}^{|P|}$, satisfying $\phi_y \geq 0$ and $\sum_{y=1}^{|P|} \phi_y = 1$, is a design parameter. Let y be the selected path for a probe and x an indicator that the probe successfully reaches its destination. Then the observation model becomes:

$$f(x,\, y;\, \boldsymbol{\theta},\, \boldsymbol{\phi}) = \phi_y(\prod_{l \in y} \theta_l)^x (1 - \prod_{l \in y} \theta_l)^{1-x}. \tag{5.1}$$

We use the MLE proposed in [25] for unicast as follows,

Proposition 18. *[25] If the measurement matrix A has full column rank and there is at least one successful probe per path, then the MLE for loss tomography is*[1]*:*

$$\hat{\boldsymbol{\theta}} = \exp\left((A^T A)^{-1} A^T \log \hat{\boldsymbol{\alpha}}\right), \tag{5.2}$$

where $\hat{\boldsymbol{\alpha}}$ is the vector of empirical path success rates.

5.2.3 Observation model and MLE of multicast

For multicast probing, the observation model is more complicated. Let $\mathbf{X} = (X_i)_{i \in V}$ denote the indicators for a multicast probe to reach individual nodes in the tree, $X_i = 0$ if the probe doesn't reach node i and $X_i = 1$ if it does. $\mathbf{X}_R = (X_i)_{i \in R}$

[1]For ease of presentation, we use $g(\mathbf{z})$ to denote the vector obtained by applying a scalar function $g(\cdot)$ to each element of vector $\mathbf{z}$.

denotes the subset of indicators for leaf nodes, and $\mathbf{X}_I = (X_i)_{i \notin R}$ the subset of indicators for internal nodes (including the root, for which $X_0 := 1$); note that only $\mathbf{X}_R$ is observable.

Since only a fraction of the X_i's are observable, the likelihood function is a marginal conditional distribution:

$$\mathbb{P}\left(\mathbf{X}_R|\boldsymbol{\theta}\right) = \sum_{\mathbf{X}_I} \mathbb{P}\left(\mathbf{X}_I, \mathbf{X}_R|\boldsymbol{\theta}\right) \tag{5.3}$$

where $p(\mathbf{X}|\boldsymbol{\theta})$ is the joint conditional distribution of all X_i's for given link success rates $\boldsymbol{\theta}$. Each $X_i, i \in V$ only depends on its parent $X_{f(i)}$, so we can write the joint distribution as,

$$\mathbb{P}\left(\mathbf{X}|\boldsymbol{\theta}\right) = \prod_{k \in V} \mathbb{P}\left(X_k|X_{f(k)}, \boldsymbol{\theta}\right). \tag{5.4}$$

Note that because of the tree structure, each $X_i, i \in V$ appears only once in front of the conditional sign "|" in (5.4).

$\mathbb{P}\left(() \mathbf{X}_k|X_{f(k)}, \boldsymbol{\theta}\right)$ is the observation model at each node k given the observation at its parent node $f(k)$. If $X_{f(k)} = 1$, then at node k, $X_k = 1$ or 0 with probability θ_k or $\bar{\theta}_k := 1 - \theta_k$ (recall that θ_k is the success probability of link k that connects nodes $f(k)$ and k); if $X_{f(k)} = 0$, then $X_k = 0$. Therefore, we have

$$\begin{aligned} \mathbb{P}\left(X_k|X_{f(k)},\, \boldsymbol{\theta}\right) = X_{f(k)}(X_k\theta_k + (1 - X_k)\bar{\theta}_k) \\ + (1 - X_{f(k)})(1 - X_k). \end{aligned} \tag{5.5}$$

Substituting (5.5) into (5.4) and then into (5.3) gives an explicit expression for the likelihood function for multicast probing.

An indirect form of the MLE is provided in [10] as follows. For each node k, define γ_k as the probability that any receiver under node k receives a multicast probe, and

a_k the probability that a multicast probe reaches node k. Then θ_k, γ_k, and a_k are related as follows:

$$\theta_k = a_k / a_{f(k)}, \quad k \in V \setminus \{0\} \tag{5.6}$$

$$a_k = \gamma_k, \quad k \in R, \tag{5.7}$$

$$1 - \gamma_k / a_k = \prod_{j \in d(k)} \left(1 - \gamma_j / a_k\right), \quad k \notin R. \tag{5.8}$$

Equations (5.6–5.8) jointly define a transformation from $\boldsymbol{\gamma}$ to $\boldsymbol{\theta}$. Note that the empirical value of γ_k, namely $\hat{\gamma}_k$:=fraction of multicast probes that are received by at least one of the receivers under node k, is directly measurable. Moreover, $\hat{\gamma}_k$ is the MLE of γ_k. If the transformation from $\boldsymbol{\gamma}$ to $\boldsymbol{\theta}$ is one-to-one, then we can easily obtain the MLE of $\boldsymbol{\theta}$ from $\hat{\boldsymbol{\gamma}}$ by applying the invariance property of MLE. Indeed, this has been shown in [10].

Theorem 19 ([10])**.** *The transformation from $\boldsymbol{\gamma}$ to $\boldsymbol{\theta}$ defined by (5.6–5.8) is a bijection. Therefore, $\hat{\boldsymbol{\theta}}$ defined by substituting $\hat{\boldsymbol{\gamma}}$ into (5.6–5.8) is the MLE of $\boldsymbol{\theta}$ under multicast probing.*

Note that the estimators defined in Equation (5.2) and Equations (5.6–5.8) are both asymptotically unbiased.

5.3 Identifiability and Path Construction

In order to compare the unicast and multicast approaches, it is important that both methods identify all of the links. Multicast probing is shown to identify all links in trees without degree-2 nodes (i.e., all nodes are either leaves or branching nodes). Below we will show that with suitably constructed paths, unicast probing can achieve the same identifiability using the same set of measurement nodes ($s \cup R$ in the multicast tree).

Algorithm 4: Unicast Path Construction for Tree Topology

1: $P \leftarrow \emptyset$
2: **for** each leaf $v \in R$ **do**
3: $\quad P \leftarrow P \cup p_{0 \to v}$
4: **end for**
5: **for** each branching node $v \in V \setminus \{0 \cup R\}$ **do**
6: $\quad$ select c_1^v, $c_2^v \in d(v)$ such that $c_1^v \neq c_2^v$
7: $\quad$ select leaves l_1^v under c_1^v and l_2^v under c_2^v
8: $\quad P \leftarrow P \cup p_{l_1^v \to l_2^v}$
9: **end for**

We establish the above by specifying a path construction algorithm that achieves identifiability. Consider Algorithm 4, which constructs a set of $|L|$ simple paths in two steps:

1. select all paths from root to leaves (lines 2–4);
2. for each branching node v, select a path between two arbitrary leaves under different children of v (lines 5–9).

Here we use the notation $p_{v \to w}$ to denote the (unique) path in the tree between nodes v and w. For example, for the multicast tree in Fig. 5.1, Step (1) constructs paths $p_{0 \to 4}$, $p_{0 \to 5}$, $p_{0 \to 6}$, and $p_{0 \to 7}$, and Step (2) constructs paths $p_{4 \to 5}$, $p_{4 \to 7}$, and $p_{6 \to 7}$.

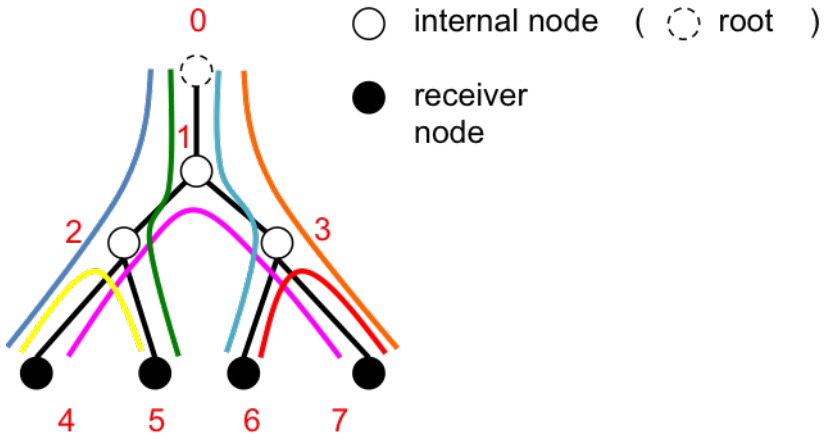


Figure 5.1: Example: unicast paths for identifying links in a multicast tree.

The following theorem states that paths constructed by Algorithm 4 identify all links in a tree without degree-2 nodes.

Theorem 20. *For a tree $\mathcal{T}$ with no degree-2 nodes, unicast probing between the monitors identifies all links in the tree if and only if all degree 1 nodes are monitors.*

Proof. It is easy to see that the condition is necessary, as otherwise the link leading to a non-monitor leaf cannot be measured. The proof of necessity follows because given the metrics of the paths constructed by Algorithm 4 each equals the sum of the traversed link metrics, thus allowing us to identify the metrics of all the links in the tree. Since, under the assumption of independent losses for unicast, link/path success rates can be converted to additive metrics by taking the logarithm, the result follows.

Let $w_{i,j}$ denote the metric of the path between nodes i and j; Given the path construction of Algorithm 4, $w_{i,j}$ can be directly estimated from path performance (e.g., losses) if and only if both i and j are degree-1 nodes (i.e., root or leaves). Suppose we construct paths according to Algorithm 4. The constructed paths have the property that they can identify $w_{0,v}$ for all v. In particular, if v is a branching node (e.g., node 1 in Fig. 5.1), and the path constructed for v in Step (2) is between leaves v_1 and v_2 (nodes 4 and 7 in Fig. 5.1), then $w_{0,v} = (w_{0,v_1} + w_{0,v_2} - w_{v_1,v_2})/2$. The metric of link $(i,\ j)$ is then determined by $w_{i,j} = w_{0,j} - w_{0,i}$, assuming node i is closer to node 0 than node j. □

Remark: Note that the theorem only states that there exists a set of unicast probing paths that suffice to identify all links. This set may not be unique, and one can add paths to the set without affecting identifiability. Which set of paths to use and how many probes to send on each path are to be optimized by experiment design.

5.4 Performance Bound and Experiment Design

Given the observation model $f(\boldsymbol{O};\theta)$ where $\boldsymbol{O}$ represents the observations, the (per-measurement) FIM wrt $\boldsymbol{\theta}$ is an $|L| \times |L|$ matrix, whose $(i,\ j)$-th entry is defined by

$$I(i,\ j) = -\mathbb{E}\left[\frac{\partial^2}{\partial\theta_i \partial\theta_j} \log f(\boldsymbol{O};\ \boldsymbol{\theta}) \Big| \boldsymbol{\theta}\right]. \tag{5.9}$$

For unicast $\boldsymbol{O} = \{x, y\}$ in (5.1), and for multicast $\boldsymbol{O} = \mathbf{X}_R$ in (5.3).

The significance of the FIM is that it provides a fundamental bound on the error of unbiased estimators. Specifically, if $\hat{\boldsymbol{\theta}}$ is an unbiased estimator of $\boldsymbol{\theta}$ using N i.i.d. measurements, then the covariance matrix of $\hat{\boldsymbol{\theta}}$ satisfies[2] $\text{cov}(\hat{\boldsymbol{\theta}}) \succeq \frac{1}{N} I^{-1}(\boldsymbol{\theta};\boldsymbol{\phi})$, known as the *Cramér-Rao bound (CRB)* [41]. In particular, the MSE in estimating θ_l, given by $\text{cov}(\hat{\boldsymbol{\theta}})_{l,l}$, is lower bounded by $I^{-1}_{l,l}(\boldsymbol{\theta};\ \boldsymbol{\phi})/N$.

5.4.1 FIM Based Experiment Design for Unicast

Based on the observation model (5.1), as shown in [25], the (i,j)-th entry of the FIM for unicast loss tomography is:

$$I_{i,j}(\boldsymbol{\theta};\boldsymbol{\phi}) = \sum_{y=1}^{|P|} \phi_y \frac{\alpha_y(\boldsymbol{\theta})}{\theta_i \theta_j (1-\alpha_y(\boldsymbol{\theta}))} \mathbb{1}\{i,j \in p_y\}. \tag{5.10}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function, and α_y the path success rate of y.

Based on the FIM, the goal of experiment design is to optimize some function of the FIM, which is related to bounding estimation errors, by choosing the design parameter $\boldsymbol{\phi}$. We leverage the previous results on optimal experiment design [25]. We consider weighted A-optimality, which is to minimize the weighted trace $\text{Trace}(I^{-1} \cdot \text{diag}(\boldsymbol{\omega}))$ with $\boldsymbol{\omega} = (\omega_l)_{l\in L}$ denoting the link weights, as it directly corresponds to

[2]For matrices A and B, $A \succeq B$ means that $A - B$ is positive semi-definite.

weighted link MSE for unbiased estimators. The previous result states that if a set of paths forms a basis for the link space, i.e. the measurement matrix A (defined in Section II.B) is invertible, then the optimal probe allocation is given by $\phi_y = \frac{\sqrt{\mathcal{A}_y(\boldsymbol{\theta},\boldsymbol{\omega})}}{\sum_{i=1}^{|L|}\sqrt{\mathcal{A}_i(\boldsymbol{\theta},\boldsymbol{\omega})}}$, where $\mathcal{A}_i(\boldsymbol{\theta},\boldsymbol{\omega})$ for path i is a function of $\boldsymbol{\theta}$ and $\boldsymbol{\omega}$. Our focus is therefore on selecting the optimal basis that optimizes the overall design objective, i.e., the trace of the inverse FIM. Under optimal probe allocation, this design objective equals $(\sum_{i=1}^{|L|}\sqrt{\mathcal{A}_i})^2$, where the coefficients $\mathcal{A}_i$ implicitly depend on the probing paths. To optimize path construction, we first derive a closed-form expression for $\mathcal{A}_i$ that explicitly depends on the decision variables in path construction.

Consider the path construction in Algorithm 4. Let p_v denote the path associated with node v: if v is a leaf, $p_v = p_{0\to v}$; if v is a branching node, $p_v = p_{l_1^v \to l_2^v}$ for the selected leaves l_1^v and l_2^v under different children of v. Note that for a given tree with a given root, the decision variables for this algorithm are $\{l_1^v,\, l_2^v : \forall \text{ branching node } v\}$.

The key to deriving an explicit expression for $\mathcal{A}_i(\boldsymbol{\theta})$ is to derive an explicit expression for the inverse measurement matrix $A^{-1} = [b_{k,i}]_{k,i=1}^{|L|}$. Let $w_{i,j}$ denote the metric of the path segment between nodes i and j, and m_v denote the end-to-end metric of path p_v. We have that the metric of link k equals the inner product between the k-th row of A^{-1} and the vector of path metrics, i.e., $w_{f(k),k} = \sum_{i=1}^{|L|} b_{k,i} m_i$. Therefore, we can obtain $b_{k,i}$ by expressing $w_{f(k),k}$ as a function of m_i's. Specifically, we can express the metric of each 0-to-v path segment as

$$w_{0,v} = \begin{cases} m_v & \text{if } v \in R, \\ \frac{m_{l_1^v} + m_{l_2^v} - m_v}{2} & \text{if } v \notin R. \end{cases} \tag{5.11}$$

Based on these 0-to-v path metrics, we can identify link metrics as

$$w_{f(k),k} = \begin{cases} \frac{m_{l_1^1} + m_{l_2^1} - m_1}{2} & \text{if } k = 1, \\ \frac{m_{l_1^k} + m_{l_2^k} - m_k}{2} - \frac{m_{l_1^{f(k)}} + m_{l_2^{f(k)}} - m_{f(k)}}{2} & \\ \qquad\qquad \text{if } k > 1,\ k \notin R, & \\ m_k - \frac{m_{l_1^{f(k)}} + m_{l_2^{f(k)}} - m_{f(k)}}{2} & \text{if } k \in R. \end{cases} \tag{5.12}$$

Comparing (5.12) with the generic formula of $w_{f(k),k} = \sum_{i=1}^{|L|} b_{k,i} m_i$ gives the value of $b_{k,i}$ as in Table 5.1,

Table 5.1: value of $b_{k,i}$

$b_{k,i} =$	$k = i$	$k \neq i$
$i \leq \lvert L\rvert - \lvert R\rvert$	$-\frac{1}{2}$	$\frac{1}{2}\mathbf{1}\{k \in d(i)\}$
$i > \lvert L\rvert - \lvert R\rvert$	$1 - \frac{1}{2}\mathbf{1}\{i \in p_{f(i)}\}$	$\frac{1}{2}\mathbf{1}\{i \in p_k, i \notin p_{f(k)}\} -$ $\frac{1}{2}\mathbf{1}\{i \notin p_k, i \in p_{f(k)}\}$

where $i \in p_v$ means that node i is on path p_v. For $i > |L| - |R|$ (i.e., i is a leaf), $i \in p_v$ if and only if $v = i$ if v is also a leaf, or $i \in \{l_1^v,\ l_2^v\}$ if v is a branching node.

Substituting the above expressions for $b_{k,i}$ into the formula for $\mathcal{A}_i(\boldsymbol{\theta})$ in [25] yields the following: if $i \leq |L| - |R|$,

$$\mathcal{A}_i(\boldsymbol{\theta}) = \frac{1 - \alpha_i}{\alpha_i} \left(\frac{1}{4} \sum_{k \in i \cup d(i)} \omega_k \theta_k^2 \right); \tag{5.13}$$

if $i > |L| - |R|$,

$$\mathcal{A}_i(\boldsymbol{\theta}) = \frac{1 - \alpha_i}{\alpha_i} \left(\omega_i \theta_i^2 (\frac{1}{4} + \frac{3}{4} \mathbb{1}\{i \notin p_{f(i)}\}) + \frac{1}{4} \sum_{k \in \Phi_i} \omega_k \theta_k^2 \right), \tag{5.14}$$

where $\Phi_i := \{k \neq i :\ i \in p_k,\ i \notin p_{f(k)}\} \cup \{k \neq i :\ i \notin p_k,\ i \in p_{f(k)}\}$. One observation from the closed-form solution is that the coefficient corresponding to the path for

a branching node, given by (5.13), only depends on path selection through α_i, the success rate of this path: the larger α_i, the smaller $\mathcal{A}_i(\boldsymbol{\theta})$. This observation motivates a heuristic that when selecting l_1^v and l_2^v, we can select the leaves with the highest success rates to reach node v, and thus the path between l_1^v and l_2^v will have the highest success rate (among all paths between leaves under different children of v). This can be computed recursively.

Let λ_v denote the highest success rate for a probe sent by v to reach any leaf under v, i.e., it is the success rate of path segment $p_{v\to l_1^v}$ for l_1^v and l_2^v selected by the above heuristic. We then have:

1. if all children of v are leaves, then l_1^v and l_2^v are the leaves connected to v by the two links with the highest success rates, and $\lambda_v = \theta_{l_1^v}$;

2. if not all children of v are leaves, then l_1^v and l_2^v are the leaves with the two largest value of $\lambda_c\theta_c$ among all $c \in d(v)$, and $\lambda_v =$

5.4.2 FIM and Performance Bound for Multicast

Based on the likelihood function (5.3), we are ready to derive an explicit expression for the FIM for multicast. By definition, the (i,j)-th entry in the FIM equals $I_{i,j}(\boldsymbol{\theta}) = -\sum_{\mathbf{X}_R} \mathbb{P}\left(\mathbf{X}_R|\boldsymbol{\theta}\right) \frac{\partial^2}{\partial\theta_i\partial\theta_j}\mathcal{L}(\mathbf{X}_R|\boldsymbol{\theta})$, where $\mathcal{L}(\mathbf{X}_R|\boldsymbol{\theta}) = \log(\mathbb{P}\left(\mathbf{X}_R|\boldsymbol{\theta}\right))$ is the log-likelihood function. Substituting (5.3) into the above equation gives

$$I_{i,j}(\boldsymbol{\theta}) = -\sum_{\mathbf{X}_R} \frac{1}{\mathbb{P}\left(\mathbf{X}_R|\boldsymbol{\theta}\right)} \Bigg[\mathbb{P}\left(\mathbf{X}_R|\boldsymbol{\theta}\right) \Big(\sum_{\mathbf{X}_I} \frac{\partial^2}{\partial\theta_i\partial\theta_j} \mathbb{P}\left(\mathbf{X}|\boldsymbol{\theta}\right) \Big) - \Big(\sum_{\mathbf{X}_I} \frac{\partial}{\partial\theta_i} \mathbb{P}\left(\mathbf{X}|\boldsymbol{\theta}\right) \Big) \Big(\sum_{\mathbf{X}_I} \frac{\partial}{\partial\theta_j} \mathbb{P}\left(\mathbf{X}|\boldsymbol{\theta}\right) \Big) \Bigg]. \tag{5.15}$$

Based on the explicit expression for $\mathbb{P}\left(\mathbf{X}|\boldsymbol{\theta}\right)$ given by (5.4) and (5.5), we have

$$\frac{\partial}{\partial\theta_i}\mathbb{P}\left(\mathbf{X}|\boldsymbol{\theta}\right) = X_{f(i)}(2X_i - 1) \prod_{k\neq i} \mathbb{P}\left(\mathbf{X}_k|X_{f(k)},\, \boldsymbol{\theta}\right), \tag{5.16}$$

and

$$\frac{\partial^2}{\partial\theta_i\partial\theta_j}\mathbb{P}\left(\mathbf{X}|\boldsymbol{\theta}\right) = \begin{cases} 0, & i=j, \\ X_{f(i)}(2X_i-1)X_{f(j)}(2X_j-1) & \\ \times \prod_{k\neq i,j} \mathbb{P}\left(\mathbf{X}_k|X_{f(k)},\,\boldsymbol{\theta}\right), & i\neq j. \end{cases} \tag{5.17}$$

Substituting (5.16, 5.17) into (5.15) gives the FIM for multicast probing, and $I_{ii}^{-1}(\theta)/N$ gives the lower bound on the MSE of any unbiased estimator of θ_i.

n the simple case of a single multicast tree, the design for multicast probing becomes trivial: we send batches of $|R|$ back-to-back unicast probes per batch, each following the path from the source to a receiver in the multicast tree. It has been verified in [17] that such a batch of back-to-back unicast probes can mimic observations taken under a multicast probe. Note that to ensure fair comparison with (independent) unicast probing, we consider each batch as $|R|$ probes instead of a single multicast probe.

5.5 Performance Evaluation

We compare the performance of loss tomography based on multicast or unicast probing by packet-level simulations on binary tree topologies. We build a simulator for multicast measurement and inference, using the inference algorithm from [10]. For unicast, we use the same simulator as in [25] and among its three proposed measurement allocation methods we use 'Iterative A-optimal design' which does not assume known link success rates. To simulate correlated unicast, we send batches of $|R|$ back-to-back unicast probes per batch, each following the path from the source

to a receiver in the multicast tree, assuming the packet loss realizations on each link are the same for all probes in the same batch.

To avoid degree-2 nodes we add a node to the root node of a standard binary tree and consider the added node as root node. Link success rates are randomly generated with a uniform distribution between 0.1 and 1 unless otherwise specified. This model is motivated from the RoofNet study [25], wherein link success rates were found to be well approximated with Uniform$(0.1, 1)$. For unicast, we simulate the iterative A-optimal design by estimating link success rates for every 100 probes, and updating the design parameter ϕ as in Algorithm 2 in [9]. For all topology and link settings, the experiment results are based on simulations of 30 Monte Carlo runs, each consisting of 100 of iterations of 100 probes per iteration.

It's not fair to compare the performance of multicast and unicast for the same number of probes because a unicast probe only traverses a path between one pair of degree 1 nodes while a multicast probe traverses the entire tree and thus has a much larger overhead. Thus we use the number of traversed hops as the "cost" of measurement. For multicast, number of hops traversed by each probe is the number of links in the whole tree. For unicast, it's the length of the probed end-to-end path. For correlated unicast, it's the sum of path lengths of all root-to-leaf paths in the tree.

5.5.1 Convergence Rate

The CRB gives a lower bound on MSE based on the FIM. It decreases with rate $1/\#probes$. Figure 5.2 shows the CRB and MSE versus number of probes for both multicast and unicast tomography for a 2-leaf tree, where MSE is calculated with 30 Monte Carlo runs. The average link CRB is 0.24 for multicast and 1.28 for unicast. Multicast has lower CRB and MSE here because each multicast probe gives

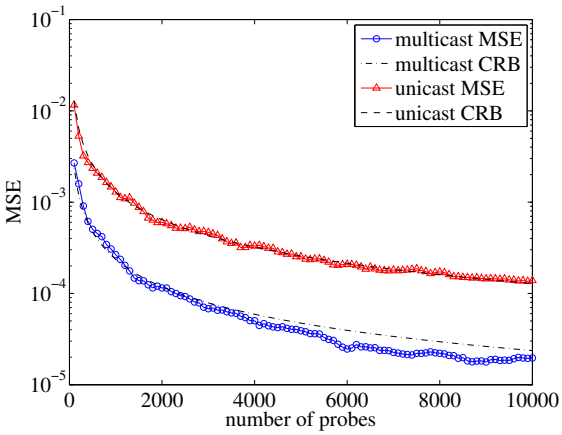


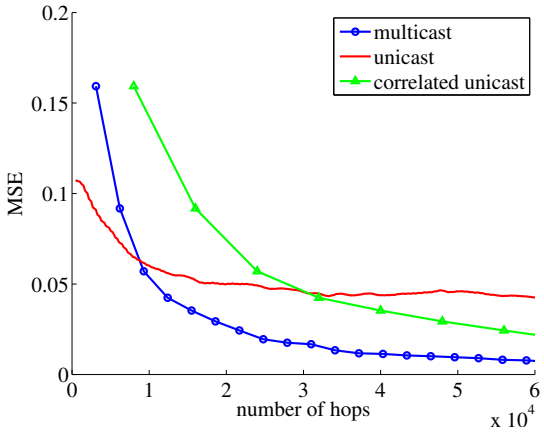
Figure 5.2: Average link MSE and CRB of a 2-leaf tree

information on all 3 links in the tree, while unicast only gives information on links in the path that's probed (each unicast probe traverses 2 links in this case).

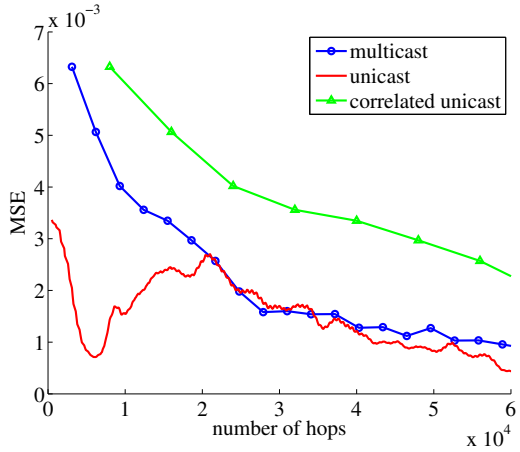
After adding plot of bias, add:) The results show that CRB is tight for both multicast and unicast even if the MLEs are biased.

5.5.2 Impact of link weights

Unicast probing has the flexibility to allocate probes unevenly across the network, which intuitively favors the cases where only a portion of the network is of interest, while multicast measures all the links evenly. We introduce link weights to model relative importance and compare the subsequent performance of all three measurement methods. Figure 5.3 shows weighted average MSE against number of hops in a 16-leaf full binary tree with 32 nodes and 31 links. For homogeneous link weights, all links have unit weight. For heterogeneous link weights, we randomly select a link and set its weight as 500 while all the other links have unit weight. When links have homogeneous weights as in Figure 5.3(a), unicast achieves a lower MSE when the number of hops is small due to its flexibility in sending more probes on paths providing more "information" as measured by the FIM. But unicast is outrun by multicast and correlated unicast as the number of hops increases. When link weights are heterogeneous



(a) homogeneous link weights: data rate = 1 Mbps, #monitors = 20



(b) heterogeneous link weights: data rate = 1 Mbps, #monitors = 20

Figure 5.3: MSE vs. number of hops with homogeneous/heterogeneous link weights. (On a full binary tree with 16 leaf nodes. All link success rates $\sim$ Uniform$(0.1, 1)$.)

as in Figure 5.3(b), the advantage of unicast becomes more significant as its probe allocation takes into account link weights.

5.5.3 Impact of link success rate distribution

In our simulation, link success rates by default are uniformly distributed between 0.1 and 1. We randomly select a fraction of links to be 'reliable links', and set their success rates to be Uniform$(0.9, 1)$. Given a fraction of reliable links, we generate 10 instances of link success rates for all links. As shown in Figure 5.4 for a full 16-

leaf binary tree, when the fraction of 'reliable links' gets larger, for a fixed and large enough (65000+) number of hops, the MSE of unicast probing becomes smaller, a similar effect was also observed in [25]. For multicast and correlated unicast probing, the MSE is largely invariant under changes in the link success rate distribution.

5.5.4 Impact of tree size

We increase the size of the full binary tree from 3 links (2 leaf nodes) to 63 links (32 leaf nodes) and evaluate the average link MSE. Figure 5.5 shows MSE vs. tree size. For a meaningful comparison among trees of difference sizes, we fix the ratio of total hop count and number of links to 2000, so that the total number of hops grows proportionally to the size of the tree. The performance of multicast and correlated unicast remains largely the same as the tree size increases, while both the median and the range of MSE of unicast probing increase as the tree grows larger.

5.6 Conclusions

We compared the performance of link loss rate inference from unicast/multicast probes using network tomography for tree topologies. We showed that both probing methods achieve identifiability when there is no degree-2 node in the tree. Our empirical comparison shows that while multicast generally gives good performance, unicast with optimized probe allocation can outperform multicast for inferring heterogeneously-weighted links under tight probing budget.

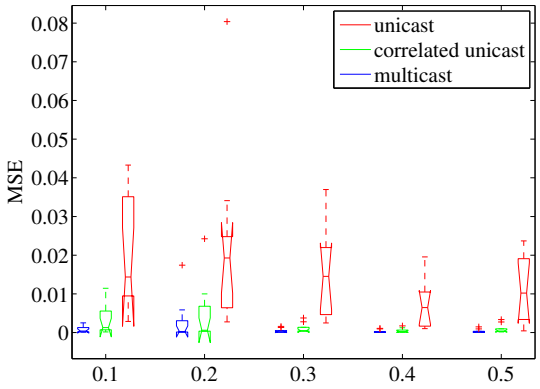


Figure 5.4: MSE vs. varying link success rate distribution (16-leaf full binary tree with homogeneous link weights)

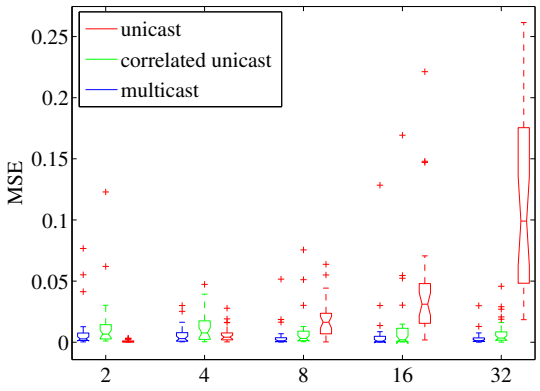


Figure 5.5: MSE vs. varying size of tree (all links have success rates $\sim$ Uniform$(0.1, 1)$ and homogeneous link weights)

CHAPTER 6

ONLINE ROUTING WITH INFERRED PATH RELIABILITY

6.1 Introduction

The performance of end-to-end connections in large scale networks is strongly affected by the performance of internal components of the network, as represented link delays or loss probabilities. Traditional routing algorithms such as OSPF determine routes between end-host pairs without considering such network internal metrics. This motivates the study and design of routing algorithms that take internal metrics such as delay or reliability into consideration for global optimal routing. When the internal metrics are know, static solutions for routing can be calculated offline such as OSPF. However, when we assume the underlying network metrics are unknown, algorithms that can learn the metrics at the same time of deciding best routes, a.k.a. online routing, are needed.

Routing takes different forms in different application scenarios. In ad hoc networks, algorithms are designed to decide on next-hop routing at each node. Another kind of routing, however, decides the whole route (that often includes multiple hops) between the source and destination node, and is referred to as end-to-end routing. This part of the thesis focuses on the latter, where we assume there are a set of paths between a source node and a destination node with unknown path reliabilities. Our goal is to design an online routing algorithm that finds the most reliable paths among them all so that given an end-to-end transmission budget, we can transmit as many packets successfully as possible from the source to the destination. We assume the

network topology is known, but link metrics(e.g. packet loss probabilities) are unknown and non-time-varying. To learn the true quality of a path, the source has to probe it repeatedly to accumulate sufficiently many samples. Although under basic stationary conditions, any probing strategy that samples each path indefinitely often (e.g. round-robin) eventually learns the true average path metrics and therefore determines the best path to use, the probing strategy affects performance over a finite time horizon, as it determines the number of times each path is measured, which then determines the quality of path estimation and thus the quality of the routes. It is hence desirable to have an online probing algorithm that not only converges to the optimal route eventually, but also minimizes the use of suboptimal routes during the convergence process. Intuitively, if all paths are independent of each other, we will have to probe each of them a sufficiently number of times in order to get accurate estimates of the underlying path metrics, which makes the problem intractable as the number of paths grows large in large scale networks. Fortunately, however, paths in our problem have strong dependencies because of the presence of many shared links. Dependencies between paths makes it possible for us to reduce the space of paths that we must probe a possibly small subset of all candidate paths that satisfies certain properties, so that we can get accurate estimates of the metrics of all path by only probing path in the subset.

Out problem fits into a classic *Multi-Armed Bandit* problem 2 setup, where the set of paths are the "arms". In the context of Multi-Armed Bandit problem, a commonly used measure of the sub-optimality of the probing strategy is *regret*, defined as the gap of the performance between the paths selected by an online algorithm and the optimal path that is the most reliable based on the true link reliabilities. The goal of this chapter is to design a online routing algorithm that determines at each discrete time steps, which path to send on a packet transmission (a.k.a. a probe), so that to

find the most reliable path while minimizing the regret (average transmission losses) during the overall process.

In the context of determining the most reliable path, we consider two observation models:

1. *Bandit feedback:* During each probe, link states are not directly observable. Sending a probe over a path results in either a success or a loss. When the latter occurs, no information is revealed as to which link might have dropped the probe

2. *Semi-bandit feedback:* During each probe, the measurement result not only includes information about whether it ends up a success or a loss, but also the location of the link where the loss occurred when a loss happens.

For each observation model, we propose an online algorithm and determines probe allocations on the set of paths at each time step. We provide a regret upper bound and regret lower bound for the bandit feedback model, and a regret upper bound the the semi-bandit feedback model. The following is a summary of our contributions:

1. *Topological structure-based learning algorithms*: We propose algorithms that leverage path reliability correlations introduced by link overlaps, and improve the performance of probing strategy by reduce the probing space from all the candidate paths to a subset of paths that satisfies certain topological properties. Specifically, we find probing the basis instead of all paths improves the performance under bandit feedback model, while probing the minimum cover set of links brings similar performance gains under the semi-bandit feedback model.

2. *Analysis of performance gain from having information on loss locality*: By analyzing the algorithms and finding their performance bounds, we compare the achievable regret under of bandit feedback model and semi-bandit feedback

model. Our results show that the potential performance gain depends on the maximum path length, d, of the candidate paths set. Let n be the number of links, when $d = O(1)$ or $d = O(\log n)$, the semi-bandit feedback model yields a smaller regret upper bound than the bandit feedback model. In the case that $d = O(1)$, the regret upper bound of semi-bandit feedback model matches the regret lower bound of the bandit feedback model.

3. *Evaluation on real traces*: We evaluate the performance of our algorithms on a small scale network with simulations. We run our algorithms to learn the paths reliability between a pair of nodes in the network. Our simulation results show that the Semi-bandit Feedback model has an obvious advantage over the Bandit Feedback model. Meanwhile for the Bandit Feedback model under small network sizes, there's no significant benefit of measuring the basis instead of all the paths. Furthermore, we investigate how the performance of our algorithms are affected by tuning some of the design parameters.

6.1.1 Related work

Learning-based routing has been studied in the wireless context and falls in two categories: hop-by-hop routing that fits especially well into ad hoc networks[6], and end host-based routing where an end host controls the selection of intermedia nodes for a route. Our work falls in the realm of end host-based routing.

Online routing has also been studied using the stochastic Multi-Armed Bandit model, where arm rewards are assumed to follow some probability distributions. Some works deal with the case where the underlying probability distributions are time-varying, while others assume the probability distributions don't change over time. Our work falls into the latter category, a.k.a., non-adversarial stochastic Multi-Armed Bandit model. When assuming arms are independent, the UCB algorithms[26] has been proposed and studied in 2012. However, routing in networks often means the

arms in the model are correlated since paths can share common links in the networks. Under this scope, people have studied various methods for online routing with regard to path delay, where the goal is to find the shortest paths. Different assumptions regarding measurement granularity have been investigated. Assuming one can observe each individual link delay when measuring a path, a method using combinatorial optimization has been proposed[21] and shown to achieve a regret upper bound of $O(n^4 \log T)$, where n is the number of links and T is the time horizen. Following the same assumption, the benefit of decoupling measurement paths and routing paths were investigated[24], and it were shown that decoupling can improve the regret upper bound by reducing upper bound to $O(\log T)$. Another line of work assumes one can only observe end-to-end delays. Utilizing the correlation of paths, an algorithm that only directly measures a subset of the paths has been proposed and proved to achieve an upper bound of regret as $O(md^3 \log T)$, where m is the number of paths and d is the maximum path length. This chapter investigates both assumptions of the measurement granularity, path-level measurement and link level measurement. And we are the first one to study online routing concerning transmission reliability.

The rest of this chapter is organized as follows. Section II formulates the problem. Section III investigates the bandit feedback model, proposes a probing algorithm and analyze its performance with a regret lower bound and regret upper bound. Section IV then investigates the semi-bandit feedback model with proposing a probing algorithm and analyzing its performance with a regret upper bound. Section V evaluates the proposed solutions. Section VI concludes the chapter.

6.2 Problem Formulation

We define the online routing problem and propose our solutions in this section. For our online routing algorithms, we provide performance bounds with proofs.

6.2.1 Network model

Let $\boldsymbol{G} = (V, L)$ be the topology of the network, where V is the set of vertices and $L = \{l_1, l_2, \cdots, l_n\}$ is the set of links. Let P be an m-by-n *path matrix*, with entries $p_{ij} = 1$ $(i = 1, 2, \ldots, m, j = 1, 2, \ldots, n)$ if link j is on path i and $p_{ij} = 0$ otherwise. Each row of P, p_i $(i = 1, 2, \ldots, m)$, is a path in the network represented as an n-dimensional row vector.

Each link $l_i \in L$ has a success probability, denoted by θ_i. In other words, $1 - \theta_i$ is the probability that a packet is lost at link l_i. We assume the true values of $\theta_i, i = 1, 2, \ldots, n$ are unknown and that packet losses are independent across links and i.i.d. over time for each link. Then the success probability of each path p_i $(i = 1, \ldots, m)$, denoted as α_i, is

$$\alpha_i = \prod_{j=1}^{n} \theta_j^{p_{ij}}.$$

Let $\vec{\theta} = (\theta_1, \theta_2, \ldots, \theta_n)^T, \vec{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_m)^T$ be the column vector of all link success probabilities and path success probabilities respectively. We have,

$$\vec{\alpha} = \exp(P \log(\vec{\theta})).$$

Without loss of generality., we assume that the path success probabilities are non-increasing in their indices, $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_m$.

6.2.2 Learning the Most Reliable Path

Our problem is that at each time step $t, t = 1, 2, \ldots, T$, we need to select a path with index $y(t)$ for one packet transmission (a.k.a. a probe). As the result of each probe, we get the reward $X(t)$, which is an indicator that equal to 1 if the probe at time t on path $y(t)$ has successfully reached the destination and 0 if the packet

is dropped somewhere along the path. Thus $\{X(t)\}$ are Bernoulli random variables with $\mathbb{P}(X(t) = 1) = \alpha_{y(t)}$.

The objective of learning the most robust path is to determine $y(t)$ for all $1 \leq t \leq T$, such that the expected cumulative reward from all time steps until T, $\mathbb{E}\left[\sum_{t=1}^{T} X(t)\right]$, is maximized.

Let p_1 denote the path with the largest success probability, and $\{X_1(t)\}$ a Bernoulli process describes packet transmission on that path with $\mathbb{P}(X_1(t) = 1) = \alpha_1$. We define the *regret* at time T as

$$\mathcal{R}(T) = \mathbb{E}\left[\sum_{t=1}^{T}[X_1(t) - X(t)]\right] \tag{6.1}$$

$$= T\mathbb{E}[X_1] - \sum_{t=1}^{T}\mathbb{E}[X(t)] \tag{6.2}$$

$$= T\alpha_1 - \sum_{t=1}^{T}\alpha_{y(t)}. \tag{6.3}$$

6.2.3 Observation models

We consider the following two observation models:

Bandit feedback model: Probing result $W(t)$ at time t consists of whether the probe has traversed the whole path successfully and the index of path,

$$W_{BF}(t) = \{X(t), y(t)\}.$$

Semi-bandit feedback model: Probing result $W(t)$ consists of the indicator $X(t)$ ($X(t) = 1$ if the probe is successful), and the index of link where the loss happens, denoted $I(t)$. In the case of a success, $I(t) = 0$. Thus we have,

$$W_{SF}(t) = \{X(t), y(t), I(t)\}.$$

6.3 Bandit feedback

In this section, we propose our solution for the Bandit Feedback model and provide performance bounds on the algorithm.

6.3.1 Path basis

Recalling we define a path $p_i, i = 1, \ldots, m$ as a n-dimensional vector where $p_{i,j} = 1$ if link l_j is on path p_i and $p_{i,j} = 0$ if not. A path basis for all the paths in P is a subset of paths so that any path outside the basis can be represented as linear combinations of paths inside the basis. There can be multiple bases for a set of paths. The minimum possible size of a basis is lower bounded by the rank of the path matrix P. Let $B = \{p_1, p_2, \ldots, p_{n_b}\}$ be the set of paths in a basis of size n_b. Then

$$p_i = \sum_{p_j \in B} c_{ij} p_j, \tag{6.4}$$

$\forall\ p_i \notin B, \exists\ \{c_{ij} :\ p_j \in B\}$ with $|c_{ij}| \leq 1$. We denote the maximum absolute value of all coefficients c_{ij} by

$$c_{\max} = \max_{p_i \notin B, p_j \in B} |c_{ij}|. \tag{6.5}$$

What motivate us to look at the basis is that the size of a basis can be much smaller than the number of all paths. If we can find the best path by focusing on learning the basis, we can significantly reduce the problem size. We present the following lemma to demonstrate this to be true for complete graphs.

Lemma 21. *In a complete graph with n nodes, there are $O((n-2)!)$ simple paths*[1] *between any pair of nodes. Furthermore, all 1-hop, 2-hop, and 3-hop paths between the pair of nodes form a basis for all the paths. The size of this basis is $O(n^2)$, which is much smaller than the total number of paths.*

[1]Simple paths are paths with no repetition of links or nodes.

Proof. Let s and d denote the source and destination nodes respectively. We prove the lemma by showing that any path p between s and d can be expressed as a linear combination of only 1-hop, 2-hop, and 3-hop paths, recalling each path is an n-dimensional vector with binary entries where the i-th entry is 1 if link i is on the path and 0 if not. It's trivial if p itself is no longer than 3 hops. If p is longer than 3 hops, let $(s, v_1, v_2, \ldots, v_k, d), k > 2$ be the sequence of nodes on the path. In the vector representation of paths, each link is corresponding to the same entry in all the paths. We denote the function that maps each link (u, v) to its index in the vector representation as $I(u, v), I : V^2 \to \{1, 2, \ldots, n\}$. We assume links are undirected, so that both (u, v) and (v, u) map to the same index, $I(u, v) = I(v, u)$. Let $p_{(v_1,v_2,\ldots,v_w)}$ be the vector representation of path $(v_1, v_2, \ldots, v_w)$, then entries $I(v_i, v_{i+1})$ of the vector with $1 \leq i \leq w - 1$ are 1's and the others entries are 0's.

The rest of proof takes two steps.

(1). We show that any link in the path that is not connected to either s or d, say, (v_i, v_j) where $v_i \neq s, v_j \neq d$, can be expressed as a linear combination of 2-hop and 3-hop paths. $p_{(v_i,v_j)}$ is a vector with only the $I(v_i, v_j)$-th element being one. Consider the following four paths (all of them are either 2-hop or 3-hop),

$$(s, v_i, d),$$

$$(s, v_j, d),$$

$$(s, v_i, v_j, d),$$

$$(s, v_j, v_i, d),$$

we have the following,

$$p_{(v_i,v_j)} = \frac{1}{2}(p_{(s,v_i,v_j,d)} + p_{(s,v_j,v_i,d)} - p_{(s,v_i,d)} - p_{(s,v_j,d)}).$$

(2). Path $p_{(s,v_1,v_2,\ldots,v_k,d)}$ can then be represented as,

$$p_{(s,v_1,v_2,\ldots,v_k,d)} = \sum_{i=1}^{k-1} p_{(v_i,v_{i+1})} + p_{(s,v_1,v_k,d)} - p_{(v_1,v_k)}.$$

Terms on the right side of the equation are either 3-hop paths or single-links that can be expressed as linear combinations of 2-hop and 3-hop paths. Thus, $p_{(s,v_1,v_2,\ldots,v_k,d)}$ can be expressed as a linear combination of 2-hop and 3-hop paths. □

6.3.2 Estimators of Path Success probabilities

Given end-to-end successes/losses on paths in the basis B, we estimate path success probabilities as follows. For a path p_i in the basis ($p_i \in B$), the estimated path success probability $\hat{\alpha}_i$ is simply its empirical success probability, i.e., the frequency that a packet successfully traverses that path. For a path p_i not in the basis ($p_i \notin B$), we use the following estimator

$$\hat{\alpha}_i = \prod_{p_j \in B} \hat{\alpha}_j^{c_{ij}}. \tag{6.6}$$

Remark: The above estimators are *maximum likelihood estimators (MLE)*. For $p_i \in B$, the empirical path success probability is the MLE of α_i as the MLE of the mean of a Bernoulli random variable is its empirical mean. For $p_i \notin B$, (6.6) is the MLE of α_i due to the invariance property of MLE, as $(\alpha_i)_{i=1}^m$ and $(\alpha_i)_{p_i \in B}$ form one-to-one mappings $\alpha_i = \prod_{p_j \in B} \alpha_j^{c_{ij}}$ $(i = 1, \ldots, m)$, where $c_{ii} = 1$ and $c_{ij} = 0$ $(p_j \in B \setminus \{i\})$ for each $p_i \in B$.

6.3.3 Algorithm LPR-BF

We propose Algorithm 5 for the online learning problem. There is a trade-off of exploration and exploitation in the online learning problem as we formulated. At each time slot, we decide either to explore a path that may not appear to be the best path in order to learn path success probabilities, or to exploit the path with the highest (empirical) success probability. To control this trade-off, we keep track of A_t, the number of time slots allocated for exploration up to time t, and explore the

Algorithm 5: Learning Path Reliability with Bandit Feedback (LPR-BF)

1: $\hat{\alpha}_i \leftarrow 0$ for each $p_i \in B$
2: $A_0 = 0$
3: **for** each timestep $t \geq 1$ **do**
4: **if** $A_{t-1} < n_b(c_b \log t + 1)$ **then**
5: // exploration:
6: $A_t \leftarrow A_{t-1} + 1$
7: explore at time t by selecting a path in the basis in a round-robin order
8: **else**
9: // exploitation:
10: $A_t \leftarrow A_{t-1}$
11: exploit at time t by selecting the path in $\{p_1, \ldots, p_m\}$ with the largest estimated success probability (breaking ties arbitrarily)
12: **end if**
13: **if** selected path is p_i for $p_i \in B$ **then**
14: update $\hat{\alpha}_i$ based on the outcome of this transmission
15: **end if**
16: **end for**

paths in the basis in a round-robin fashion whenever A_t falls below a time-dependent threshold (lines 4-12). In line 4, c_b and n_b are topology-dependent variables, where n_b is the size of B and c_b must satisfy $n_b c_b > 1$ (as required for Eq. (6.7)).

6.3.4 Performance Bounds for Algorithm 5

In the proofs we use the following version of a Chernoff bound that is directly derived from the Chernoff bound in [37]:

Lemma 22. *Let $X_1, X_2, \ldots, X_N$ be i.i.d. Bernoulli random variables with the same mean μ. And let $\bar{X} = \frac{1}{N}\sum_{i=1}^{N} X_i$ be the sample mean of size N, then for $0 < \Delta < 1$,*

$$\mathbb{P}\left(\bar{X} \geq (1+\Delta)\mu\right) \leq e^{-N\mu\Delta^2/3};$$
$$\mathbb{P}\left(\bar{X} \leq (1-\Delta)\mu\right) \leq e^{-N\mu\Delta^2/2} < e^{-N\mu\Delta^2/3}.$$

We present the following lemma, which will assist the proof of the regret upper bound of Algorithm 5.

Lemma 23. *With Algorithm 5, if $n_b c_b > 1$, we have,*

$$A_t \leq \lceil n_b(c_b \log t + 1) \rceil, \forall t \geq 1. \tag{6.7}$$

Furthermore, let[2] $t^* := \min\{t : t \in \mathbb{N}, t - 1 \geq n_b(c_b \log t + 1)\}$*, then*

$$A_t \geq n_b(c_b \log t + 1), \forall t \geq t^*. \tag{6.8}$$

Proof. In the scope of this proof, we omit the subscript in n_b and c_b, and use n and c instead.

(1) Proof of (6.7) by induction.

Since $A_0 = 0$, and $A_1 = A_0 + 1 = 1 \leq \lceil nc(\log(1) + 1) \rceil$, (6.7) holds for $t = 1$. Assume (6.7) holds for $t = t'$ so that $A_{t'} \leq \lceil n(c \log(t') + 1) \rceil$. Then for $A_{t'+1}$ we have the following,

if $A_{t'} \geq n(c \log(t' + 1) + 1)$, then

$$\begin{aligned} A_{t'+1} &= A_{t'} \\ &\leq \lceil n(c \log(t') + 1) \rceil \\ &< \lceil n(c \log(t' + 1) + 1) \rceil; \end{aligned}$$

and if $A_{t'} < n(c \log(t' + 1) + 1)$, then

$$\begin{aligned} A_{t'+1} &= A_{t'} + 1 \\ &\leq \lceil n(c \log(t' + 1) + 1) \rceil - 1 + 1 && (A_{t'} \in \mathbb{N}) \\ &= \lceil n(c \log(t' + 1) + 1) \rceil. \end{aligned}$$

Thus, by induction $A_t \leq \lceil n(c \log t + 1) \rceil, \forall t \geq 1$.

---

[2]Here $\mathbb{N}$ denotes the set of positive integers.

(2) Proof of (6.8).

By definition, $t-1 < n(c\log t+1)$ and $A_t = A_{t-1}+1, \forall t < t^*$. Because $A_1 = 1$, we have

$$A_t = t, \forall t < t^*.$$

Thus, $A_{t^*-1} = t^* - 1 \geq n(c\log t^*+1)$, and $A_{t^*} = A_{t^*-1} \geq n(c\log t^*+1)$.

By definition,

$$t^* - 1 \geq n(c\log t^*+1) \tag{6.9}$$

$$t^* - 2 < n(c\log(t^*-1)+1) \tag{6.10}$$

Subtracting (6.10) from (6.9), yields

$$1 > nc\log(t^*) - nc\log(t^*-1).$$

Let $f(x) = nc\log x + 1 - nc\log(x-1)$. Its derivative $f'(x) < 0, \forall x > 1$, which implies $f(t) < 1, \forall t \geq t^*$.

Assume (6.8) holds for $t = t' \geq t^*$ such that $A_{t'} \geq n(c\log t'+1)$. When $t = t'+1$, if $A_{t'} \geq n(c\log(t'+1)+1)$, then $A_{t'+1} \geq A_{t'} \geq n(c\log(t'+1)+1)$. Otherwise if if $A_{t'} < n(c\log(t'+1)+1)$, then

$$\begin{aligned} A_{t'+1} &= A_{t'} + 1 \\ &\geq n(c\log t'+1) + 1 \\ &> n(c\log t'+1) + [nc\log(t'+1) - nc\log(t')] \\ &= n(c\log(t'+1)+1). \end{aligned}$$

thus by induction, $A_t > n(c\log t+1), \forall t \geq t^*$. □

6.3.5 Regret Lower bound

Recall that a *uniformly good* algorithm is one that provides consistent performance no matter what the underlying arm distributions are. Here we derive a regret lower bound on any *uniformly good* algorithm, based on the following theorem from [30],

Theorem 24. *[30] For any uniformly good algorithm, and suboptimal arm i s.t. $\mathbb{E}[X_i] < \mathbb{E}[X^*]$, we have:*

$$\liminf_{T\to\infty} \frac{\mathbb{E}[\tau_i(T)]}{\log(T)} \geq \frac{1}{D(X_i||X^*)}, \tag{6.11}$$

where $\tau_i(T)$ is the total number of times that the algorithm pulls arm i up to time T, and $D(\cdot||\cdot)$ is the KL distance.

We provide a regret lower bound for bandit feedback model that holds under any topology and path success probability configuration. We begin with the following definition.

Intuitively, paths in the basis are critical where erroneous success probability estimates can lead to suboptimal routing decisions. The cost for reducing such errors so that the algorithm can distinguish the best path from suboptimal paths in the basis is a lower bound on the cost to distinguish the best path from all suboptimal paths, simply because $P_B \subseteq P$. Since path measurement outcomes are random variables, we quantify the error by the Kullback-Liebier (KL) distance. For each non-optimal path $p_i \in P_B, i \neq 1$, there is a minimum error in the path success probability distribution to make a suboptimal path optimal, defined as

$$D_i \triangleq D(\alpha_i||\alpha_1), \tag{6.12}$$

where $D(\cdot||\cdot)$ is the KL distance and α_1 is the path success probability of the optimal path.

We have the following regret lower bound for any uniformly good algorithm that has an $O(\log T)$ regret upper bound.

Theorem 25. *For mutually independent and i.i.d. link success probabilities, the average regret of of any $O(\log T)$-regret algorithm under bandit feedback satisfies*

$$R(T) \geq \sum_{p_i \in P_B, i \neq 1} \frac{\Delta_i}{D_i} \log T, \tag{6.13}$$

for all sufficiently large T, where $\Delta_i \stackrel{\Delta}{=} \alpha_1 - \alpha_i$ is the expected regret of probing sub-optimal path p_i once. Hence,

$$R(T) = \Omega(n_b \log T).$$

Proof. Based on Theorem 24, we know the expected number of times a sub-optimal path $p_i \in P_B, i \neq 1$ in the basis is probed is bounded from below as follows,

$$\lim_{T \to \infty} \frac{\mathbb{E}\left[\tau_i(T)\right]}{\log T} \geq \frac{1}{D_i}. \tag{6.14}$$

The overall regret then,

$$\begin{aligned} R(T) &= \mathbb{E}\left[\sum_{i \neq 1} \delta_i \tau_i(T)\right] \\ &\geq \mathbb{E}\left[\sum_{i \in B, i \neq 1} \Delta_i \tau_i(T)\right] \\ &= \sum_{i \in B, i \neq 1} \Delta_i \mathbb{E}\left[\tau_i(T)\right] \\ &\geq \sum_{i \in B, i \neq 1} \frac{\Delta_i}{D_i} \log T, \end{aligned}$$

for sufficiently large T. □

6.3.6 Regret upper bound

Before we present the upper bound on the regret for Algorithm LPR-BF, we need to first introduce the following notations,

$$\begin{aligned}\delta_i &\triangleq (\alpha_1 - \alpha_i)/2, i = 2, 3, \ldots, n;\\ \delta &\triangleq (\alpha_1 - \alpha_2)/2 = \delta_2;\\ \delta_{min} &\triangleq 1 - (1+\delta)^{\frac{-1}{n_b c_{\max}}}.\end{aligned}$$

Here δ_i is half the difference between the success probabilities of path p_i and the best path. δ is the minimum of all $\delta_i (i \neq 1)$. In this work, we assume $\exists a > 0$ s.t. $\delta \geq a > 0$ is bounded away from 0. Let us examineδ_{min}.Note that δ_{min} is a function of $n_b, c_{\max}, \delta$, and

$$\lim_{n_b \to \infty} \frac{\delta_{min}}{1/n_b} = \lim_{n_b \to \infty} \frac{-\frac{1}{n_b^2 c_{\max}}(1+\delta)^{\frac{-1}{n_b c_{\max}}} \log(1+\delta)}{-\frac{1}{n_b^2}} = \frac{1}{c_{\max}}.$$

Thus,

$$\delta_{min} = \Theta\left(\frac{1}{c_{\max} n_b}\right). \tag{6.15}$$

Now we introduce the following upper bound on the regret.

Theorem 26. *Assume the individual link success probabilities are bounded away from 0, c_b in Algorithm LPR-BF satisfies $c_b > 3/(\alpha_m \delta_{min}^2) = \Theta(3c_{max}^2 n_b^2 \alpha_m^{-1})$ where α_m is the minimum path success probability of all paths, and t^* is defined as in Lemma 23. Then the regret at time T is bounded by*

$$\begin{aligned}\mathcal{R}(T) &\leq \lceil n_b(c_b \log(T) + 1) \rceil + (m - n_b) n_b (1 + \frac{1}{\frac{1}{3} c_b \alpha_m \delta_{min}^2 - 1})\\ &\quad + n_b (1 + \frac{1}{\frac{1}{3} c_b \alpha_m \delta^2 - 1}) + t^*\\ &= O((\frac{1}{\theta_{min}})^d n_b^3 \log T),\end{aligned}$$

where d is the maximum path length.

Proof. The regret at time T, $\mathcal{R}(T)$ comes from two mutually exclusive parts of the algorithm,

$$\mathcal{R}(T) = \mathcal{R}_A(T) + \mathcal{R}_I(T)$$

where $\mathcal{R}_A(T)$ is the regret during exploration, and $\mathcal{R}_I(T)$ is the regret during exploitation.

(1). Regret during exploration

By Lemma 23, the total number of probes for exploration is bounded by $A_T \leq \lceil n_b(c_b \log T + 1)\rceil$, and because the regret of a single probe is less than or equal to one, obviously $\mathcal{R}_A(T) \leq \lceil n_b(c_b \log T + 1)\rceil$.

(2). Regret during exploitation

Let $\tau_i(t)$ be the number of times that path p_i is probed until time t. For all $t \geq t^*$ and $i \in B$, Lemma 23 implies that $A_t > n_b c_b \log t$. Thus $\tau_i(t) \geq \lfloor A_t/n_b \rfloor \geq c_b \log t$, which means each path in the basis is probed at least $c_b \log t$ times during exploration (there could be more probes during exploitation). Now we look at the probability that any suboptimal path is selected for exploitation. Let $E_i(t)$ denote the event that a suboptimal path $p_i (i \neq 1)$ is selected for exploitation at time t. Recall that $\delta_i = (\alpha_1 - \alpha_i)/2$ $(i \geq 2)$. We have

$$\begin{aligned}\mathbb{P}\left(E_i(t)\right) &\leq \mathbb{P}\left(\hat{\alpha}_i(t) \geq \hat{\alpha}_1(t)\right) \\ &\leq \mathbb{P}\left(\{\hat{\alpha_1}(t) \leq \alpha_1 - \delta_i\} \cup \{\hat{\alpha_i}(t) \geq \alpha_i + \delta_i\}\right).\end{aligned}$$

Let $E(t) = \cup_{i\neq 1} E_i(t)$ be the event that any suboptimal path is selected for exploitation at time t. Since $\delta \leq \delta_i$ $(i \neq 1)$, we have

$$\begin{aligned}\mathbb{P}\left(E(t)\right) &= \mathbb{P}\left(\cup_{i\neq 1} E_i(t)\right) \\ &\leq \mathbb{P}\left(\hat{\alpha_1}(t) \leq \alpha_1 - \delta\right) + \sum_{i\neq 1} \mathbb{P}\left(\hat{\alpha_i}(t) \geq \alpha_i + \delta\right). \qquad (6.16)\end{aligned}$$

Now we focus on $\mathbb{P}\left(\hat{\alpha}_i(t) \geq \alpha_i + \delta\right)$ for $i \in B$ and $i \notin B$ respectively. For $i \notin B$,

$$\begin{aligned}
&\mathbb{P}\left(\hat{\alpha}_i(t) \geq \alpha_i + \delta\right) \leq \mathbb{P}\left(\hat{\alpha}_i(t) \geq (1+\delta)\alpha_i\right) \\
&= \mathbb{P}\left(\prod_{j\in B} \hat{\alpha}_j(t)^{c_{ij}} \geq (1+\delta)\prod_{j\in B} \alpha_j^{c_{ij}}\right) \\
&\leq \mathbb{P}\left(\cup_{j\in B}\{\hat{\alpha}_j(t)^{c_{ij}} \geq (1+\delta)^{\frac{1}{n_b}}\alpha_j^{c_{ij}}\}\right) \\
&\leq \sum_{j\in B} \mathbb{P}\left(\hat{\alpha}_j(t)^{c_{ij}} \geq (1+\delta)^{\frac{1}{n_b}}\alpha_j^{c_{ij}}\right) \\
&= \sum_{j\in B, c_{ij}>0} \mathbb{P}\left(\hat{\alpha}_j(t) \geq (1+\delta)^{\frac{1}{n_b c_{ij}}}\alpha_j\right) \\
&\quad + \sum_{j\in B, c_{ij}<0} \mathbb{P}\left(\hat{\alpha}_j(t) \leq (1+\delta)^{\frac{1}{n_b c_{ij}}}\alpha_j\right) \\
&\leq \sum_{j\in B, c_{ij}>0} \mathbb{P}\left(\hat{\alpha}_j(t) \geq (1+\delta)^{\frac{1}{n_b c_{\max}}}\alpha_j\right) \\
&\quad + \sum_{j\in B, c_{ij}<0} \mathbb{P}\left(\hat{\alpha}_j(t) \leq (1+\delta)^{\frac{-1}{n_b c_{\max}}}\alpha_j\right) \qquad (6.17)
\end{aligned}$$

recall that $c_{\max} = \max_{i,j}|c_{ij}|$. Because $\hat{\alpha_j}(t)$ $(j \in B)$ is the sample mean of $\tau_j(t)$ samples, we can apply the Chernoff bound as follows

$$\begin{aligned}
\mathbb{P}\left(\hat{\alpha}_j(t) \geq (1+\delta)^{\frac{1}{n_b c_{\max}}}\alpha_j\right) &\leq e^{-\tau_j(t)\alpha_j((1+\delta)^{\frac{1}{n_b c_{\max}}}-1)^2/3} \\
&\leq e^{-c_b \log(t)\alpha_j((1+\delta)^{\frac{1}{n_b c_{\max}}}-1)^2/3} \\
&= t^{-c_b\alpha_j((1+\delta)^{\frac{1}{n_b c_{\max}}}-1)^2/3} \\
&\leq t^{-c_b\alpha_m((1+\delta)^{\frac{1}{n_b c_{\max}}}-1)^2/3}, \qquad (6.18)
\end{aligned}$$

where α_m is the success probability of the worst path. Similarly,

$$\mathbb{P}\left(\hat{\alpha}_j(t) \leq (1+\delta)^{\frac{-1}{n_b c_{\max}}}\alpha_j\right) \leq t^{-c_b\alpha_m(1-(1+\delta)^{\frac{-1}{n_b c_{\max}}})^2/3}. \qquad (6.19)$$

Note that

$$\delta_{min} = 1 - (1+\delta)^{\frac{-1}{n_b c_{\max}}},$$

and that

$$1-(1+\delta)^{\frac{-1}{n_b c_{\max}}} \leq (1+\delta)^{\frac{1}{n_b c_{\max}}} - 1,$$

we can substitute (6.18-6.19) into (6.17) to obtain

$$\mathbb{P}\left(\hat{\alpha}_i(t) \geq \alpha_i + \delta\right) \leq n_b t^{-c_b \alpha_m \delta_{min}^2/3}. \tag{6.20}$$

For $i \in B$, we have

$$\begin{aligned}
\mathbb{P}\left(\hat{\alpha}_i(t) \geq \alpha_i + \delta\right) &\leq \mathbb{P}\left(\hat{\alpha}_i(t) \geq \alpha_i(1+\delta)\right) \\
&\leq e^{-c_b \log(t) \alpha_i \delta^2/3} \\
&= t^{-c_b \alpha_i \delta^2/3} \\
&\leq t^{-c_b \alpha_m \delta^2/3}.
\end{aligned} \tag{6.21}$$

Similarly, we can bound $\mathbb{P}\left(\hat{\alpha_1}(t) \leq \alpha_1 - \delta\right)$. Again, we need to separate the cases of $1 \in B$ and $1 \notin B$. Following the same argument as in (6.21), we obtain for $1 \in B$,

$$\mathbb{P}\left(\hat{\alpha_1}(t) \leq \alpha_1 - \delta\right) \leq t^{-c_b \alpha_m \delta^2/3}. \tag{6.22}$$

For the case of $1 \notin B$,

$$\begin{aligned}
&\mathbb{P}\left(\hat{\alpha_1}(t) \leq \alpha_1 - \delta\right) && (6.23) \\
&\leq \sum_{j \in B, c_{1j} > 0} \mathbb{P}\left(\hat{\alpha}_j(t) \leq (1-\delta)^{\frac{1}{n_b c_{\max}}} \alpha_j\right) \\
&\quad + \sum_{j \in B, c_{1j} < 0} \mathbb{P}\left(\hat{\alpha}_j(t) \geq (1-\delta)^{\frac{-1}{n_b c_{\max}}} \alpha_j\right) \\
&\leq n_b t^{-c_b \alpha_m \delta_{min}^2/3}. && (6.24)
\end{aligned}$$

Substituting the bounds (6.20, 6.21, 6.22, 6.24) into (6.16) yields for all $t \geq t^*$,

$$\mathbb{P}\left(E(t)\right) \leq (m - n_b) n_b t^{-c_b \alpha_m \delta_{min}^2/3} + n_b t^{-c \alpha_m \delta^2/3}. \tag{6.25}$$

Summing up (6.25) over all $t \geq t^*$ bounds the regret during exploitation as follows

$$\mathcal{R}_I(T) = \sum_{t=1}^{T} \mathbb{P}\left(E(t)\right)\left(\alpha_1 - \alpha_{y(t)}\right) \tag{6.26}$$

$$\leq \sum_{t=1}^{T} \mathbb{P}\left(E(t)\right) \tag{6.27}$$

$$\leq t^* + \sum_{t=t^*}^{T} [(m-n_b)n_b t^{-c_b \alpha_m \delta_{min}^2/3} + n_b t^{-c_b \alpha_m \delta^2/3}] \tag{6.28}$$

$$\leq t^* + (m-n_b)n_b \sum_{t=1}^{T} t^{-c_b \alpha_m \delta_{min}^2/3} + n_b \sum_{t=1}^{T} t^{-c_b \alpha_m \delta^2/3}. \tag{6.29}$$

Since for any $x > 1$, $\sum_{t=1}^{\infty} t^{-x} \leq 1 + 1/(x-1)$, we know that for any $c_b > [\alpha_m \delta_{min}^2/3]^{-1}$, (6.29) converges to a finite value as $T \to \infty$, upper bounded by

$$\mathcal{R}_I(T) \leq t^* + (m-n_b)n_b\left(1 + \frac{1}{\frac{1}{3}c_b\alpha_m\delta_{min}^2 - 1}\right) + n_b\left(1 + \frac{1}{\frac{1}{3}c_b\alpha_m\delta^2 - 1}\right),$$

which is constant in T. Note that t^* is a constant that depends only on $n_b c_b$.

(3). Combining $R_A(T)$ and $R_I(T)$

Together with the bound on $\mathcal{R}_a(T)$, we have as $T \to \infty$,

$$\begin{aligned}\mathcal{R}(T) &= \mathcal{R}_a(T) + \mathcal{R}_I(T) \\ &\leq \lceil n_b(c_b \log(T) + 1)\rceil + (m-n_b)n_b\left(1 + \frac{1}{\frac{1}{3}c_b\alpha_m\delta_{min}^2 - 1}\right) \\ &\quad + n_b\left(1 + \frac{1}{\frac{1}{3}c_b\alpha_m\delta^2 - 1}\right) + t^*.\end{aligned}$$

Note that c_b must be greater than $[\alpha_m \delta_{min}^2/3]^{-1}$. Assuming link success probabilities are bounded away from 0 s.t. $\exists a > 0$ and the minimum possible link

success probability $\theta_{min} \geq a > 0$. Then $\alpha_m^{-1} \leq (\frac{1}{\theta_{min}})^d$. Furthermore, recalling $\delta_{min} = \Theta(\frac{1}{c_{\max} n_b})$, we have $c_b = \Theta(c_{\max}^2 n_b^2)$. Thus the final regret $R(T)$ is in the order of $O((\frac{1}{\theta_{min}})^d c_{\max}^2 I s h n_b^3 \log(T))$, where d is the maximum path length.

□

6.4 Semi-bandit feedback

In this section, we propose our solution for the Semi-Bandit Feedback model and provide a performance upper bound on our algorithm.

6.4.1 Link Cover

A link cover of path set P is a subset of P, denoted $P_c \subseteq P$ that covers all the links that appear in P, while a minimum link cover is a link cover with the minimum number of paths. Let $n_c = |P_c|$ denote the number of paths in link cover P_c. There can be many link covers for a set of paths. The algorithm we propose can be applied with any link cover, but with the minimum link cover it achieves the best performance bound. Finding a minimum cover set is a classic set covering problem and is proved to be NP-hard. We won't explore about how to find the minimum link cover given a set of paths, but assumes it is given.

The motivation for focusing on link cover is similar to that on focusing on the basis, namely we want to reduce the problem size by exploring a smaller set of paths instead of the set of all the paths. A link cover provides information of all the links, while its size can be much smaller than total number of paths. We present the following lemma to demonstrate this to be true for complete graphs.

Lemma 27. *In a complete graph with n nodes, there are $O((n-2)!)$ simple paths*[3] *between any pair of nodes, and 3-hop paths between the pair of nodes form a link cover*

[3]A reminder: simple paths are paths with no repetition of links or nodes.

for all the paths. The size of this cover is $O(n^2)$, which is much smaller than the total number of paths.

Proof. If we don't consider the direct link from the source node, denoted s, to the destination node, denoted d, there are three kinds of links in all paths: links connected to s, links connected to d, and links connected to neither s nor d. Any link that connects to node s, denoted link (s, u), is covered by path (s, u, w, d) where w is any node that's not s, u, or d. Similarly, any link that connects to node d, denoted (u, d), is covered by path (s, w, u, d) where w is an arbitrary node that's not s, u, or d. Any link (u, v) that's not connected to neither s or d is covered by path (s, u, v, d). Thus, the set of 3-hop paths is a link cover for all the paths between s and d. □

The set of all 3-hop paths may not be the minimum link cover, but in the case of a complete graph it is much smaller than the set of all paths.

6.4.2 Estimator

During the probing process, we maintain two counters for each link, success counter s_i for the number of successful transmissions on link i, and failure counter f_i for the number of losses that occur on link i. It's easy to update s_i and f_i using the feedback information of each probe. Recall the feedback information of each probe for the Semi-Bandit Feedback model is defined as $W_{SF}(t) = \{X(t), y(t), I(t)\}$, where $X(t)$ is the binary indicator that equals to 1 if a probe is a success and 0 if not, $y(t)$ denotes the index of the path the probe is sent on, and $I(t)$ is the location of loss ($I(t) = 0$ is the probe is successful). When $X(t) = 1$, we add 1 to the success counter s_i for all the links in the path probes. When $X(t) = 0$, to the links preceding link $l_{I(t)}$ we update their success counter by adding 1, and we add 1 to the failure counter $f_{I(t)}$ for link $l_{I(t)}$.

Algorithm 6: Learning Path Reliability with Semi-bandit feedback (LPR-SF)

$A_0 = 0$
for each each timestep $t \geq 1$ **do**
  if $A_{t-1} < n_c(c_c \log t + 1)$ **then**
    $A_t \leftarrow A_{t-1} + 1$
    explore at time t by selecting a path in the minimum cover in a round-robin order
  else
    $A_t \leftarrow A_{t-1}$
    exploit at time t by selecting the path in $\{p_1, \ldots, p_m\}$ with the largest estimated success probability (breaking ties arbitrarily)
  end if
  update $\hat{\theta}$ and $\hat{\alpha}$ after the probe
end for

After updating counters s_i and f_i for all links after each probe, we first estimate the link success probabilities with,

$$\hat{\theta}_i = \frac{s_i}{s_i + f_i}. \tag{6.30}$$

And then we estimate path success probabilities with,

$$\hat{\alpha}_i = \prod_{j=1}^{n} \hat{\theta}_j^{p_{ij}}, \tag{6.31}$$

where $p_{ij} = 1$ if link l_j is on path p_i and $p_{ij} = 0$ if not.

6.4.3 Algorithm

With a given minimum cover set P_c of the links, we propose Algorithm 6, a.k.a. Algorithm LPR-SF, for the semi-bandit feedback model.

6.4.4 Regret Upper Bound

For each link i we choose a path p from P_c that contains the shortest path to link i and call this path the primal path of the link, $p = \text{Primal}(i)$. Let $\text{Prec}(i)$ be the

set of links preceding link i in the primal path of i. The probability of getting an observation of link i from a probe on the primal path:

$$\beta_i = \prod_{j \in \text{Prec}(i)} \theta_j. \tag{6.32}$$

Furthermore, we define

$$\delta' = \min\{(1+\delta)^{\frac{1}{d}} - 1, 1 - (1+\delta)^{\frac{1}{d}}\}, \tag{6.33}$$

and

$$\eta = \min_{i \in L} \theta_i \beta_i \geq \alpha_m. \tag{6.34}$$

Theorem 28. *With Algorithm LPR-SF, for any constant c_c satisfying $c_c > [\eta \delta'^2/3]^{-1} = O((\frac{1}{\theta_{min}})^d d^2)$, the regret at time T is bounded by*

$$R(T) \leq n_c c_c \log(T) + m n_c d(1 + \frac{1}{c_c \beta_j \delta'^2 \theta_j/3 - 1}) \tag{6.35}$$

$$= O((\frac{1}{\theta_{min}})^d n_c d^2 \log(T)). \tag{6.36}$$

as time $T \to \infty$.

Proof. The regret at time T, $\mathcal{R}(T)$ comes from two mutually exclusive parts of the algorithm,

$$\mathcal{R}(T) = \mathcal{R}_A(T) + \mathcal{R}_I(T)$$

where $\mathcal{R}_A(T)$ is the regret during exploration, and $\mathcal{R}_I(T)$ is the regret during exploitation.

With Lemma 23, the total number of probes for exploration is bounded by $A_T \leq \lceil n_c(c_c \log T + 1) \rceil$. Thus, the regret accumulated from the exploration phase is,

$$\begin{aligned} R_A(T) &\leq (\alpha_1 - \alpha_m) \times A_T \\ &\leq 1 \times n_c(c_c \log(T) + 1) \\ &= n_c(c_c \log(T) + 1). \end{aligned}$$

Now we analyze the regret cumulated during the exploitation phase.

$$\begin{aligned} R_I = &\sum_{t=1}^{T} \mathbb{P}\left(E(t)\right) \Delta_m \leq \sum_{t=1}^{T} \mathbb{P}\left(E(t)\right) \\ &\leq \sum_{t=1}^{T} \sum_{i=2}^{m} \mathbb{P}\left(E_i(t)\right), \end{aligned}$$

where $E_i(t)$ denotes the event that path i is selected for exploitation at time t instead the best path p_1. Following the same argument, Equation (6.16) holds for the semi-bandit feedback as well, namely

$$\mathbb{P}\left(E(t)\right) = \mathbb{P}\left(\cup_{i \neq 1} E_i(t)\right) \tag{6.37}$$

$$\leq \mathbb{P}\left(\hat{\alpha_1}(t) \leq \alpha_1 - \delta\right) + \sum_{i \neq 1} \mathbb{P}\left(\hat{\alpha_i}(t) \geq \alpha_i + \delta\right). \tag{6.38}$$

For $i \geq 2$,

$$\begin{aligned} \mathbb{P}\left(\hat{\alpha_i}(t) \geq \alpha_i + \delta\right) &\leq \mathbb{P}\left(\hat{\alpha_i}(t) \geq \alpha_i(1+\delta)\right) \\ &\leq \sum_{j \in p_i} \mathbb{P}\left(\hat{\theta}_j \geq \theta_j (1+\delta)^{\frac{1}{L_i}}\right). \end{aligned}$$

And for the best path $i = 1$,

$$\begin{aligned}\mathbb{P}\left(\hat{\alpha_1}(t) \geq \alpha_1 - \delta\right) &\leq \mathbb{P}\left(\hat{\alpha_1}(t) \geq \alpha_1(1-\delta)\right) \\ &\leq \sum_{j \in p_1} \mathbb{P}\left(\hat{\theta}_j \geq \theta_j (1-\delta)^{\frac{1}{L_1}}\right).\end{aligned}$$

We use the following Chernoff bound to bound the probability that link success probability estimate $\hat{\theta}_j$ exceeds θ_j,

$$\mathbb{P}\left(\hat{\theta}_j \geq \theta_j(1+\delta') | s_j = k\right) \leq e^{-k\delta'^2\theta_j/3}. \tag{6.39}$$

Meanwhile the number of observations on a link follows a binomial distribution, $s_{\text{Primal}(j)} \sim \text{Binom}(\tau_{\text{Primal}(j)}, \beta_j)$. According to Lemma 23, $\tau_{\text{Primal}(j)} \geq c_c \log(t)$, thus

$$\mathbb{P}\left(s_j \leq (1-\delta_0) c_c \log(t) \beta_j\right) \leq e^{-c_c \log(t) \beta_j \delta_0^2/3} = t^{-c_c \beta_j \delta_0^2/3}. \tag{6.40}$$

Thus, we can bound the probability of over-estimating the success probability of link j as the following,

$$\begin{aligned}&\mathbb{P}\left(\hat{\theta}_j \geq \theta_j(1+\delta')\right) \\ &= \mathbb{P}\left(\hat{\theta}_j \geq \theta_j(1+\delta') | s_j > (1-\delta_0) c_c \log(t) \beta_j\right) \times \\ &\quad \mathbb{P}\left(s_j > (1-\delta_0) c_c \log(t) \beta_j\right) \\ &\quad + \mathbb{P}\left(\hat{\theta}_j \geq \theta_j(1+\delta') | s_j \leq (1-\delta_0) c_c \log(t) \beta_j\right) \times \\ &\quad \mathbb{P}\left(s_j \leq (1-\delta_0) c_c \log(t) \beta_j\right) \\ &\leq e^{-c_c \log(t) \theta_j \beta_j \delta'^2/3} (1 - t^{-c_c \beta_j \delta_0^2/3}) + t^{-c_c \beta_j \delta_0^2/3},\end{aligned}$$

where $\delta_0 > 0$ is an arbitrary constant. Similarly, the probability of under-estimating the success probability of link j has the same bound,

$$
\begin{aligned}
&\mathbb{P}\left(\hat{\theta}_j \leq \theta_j(1-\delta')\right) \\
&= \mathbb{P}\left(\hat{\theta}_j \leq \theta_j(1-\delta') | s_j > (1-\delta_0)c_c \log(t)\beta_j\right) \times \\
&\quad \mathbb{P}\left(s_j > (1-\delta_0)c_c \log(t)\beta_j\right) \\
&\quad + \mathbb{P}\left(\hat{\theta}_j \leq \theta_j(1-\delta') | s_j \leq (1-\delta_0)c_c \log(t)\beta_j\right) \times \\
&\quad \mathbb{P}\left(s_j \leq (1-\delta_0)c_c \log(t)\beta_j\right) \\
&\leq e^{-c_c \log(t)\theta_j\beta_j\delta'^2/3}(1 - t^{-c_c\beta_j\delta_0^2/3}) + t^{-c_c\beta_j\delta_0^2/3}.
\end{aligned}
$$

Recall that,

$$
\begin{aligned}
\delta' &= \min\{(1+\delta)^{\frac{1}{d}} - 1, 1 - (1+\delta)^{\frac{1}{d}}\} \\
\eta &= \min_{i \in L}[\theta_i\beta_i],
\end{aligned}
$$

and that d is the maximum path length. It can be shown that $\delta' = \Theta(1/d)$. Then the overall regret from exploitation is

$$
R_I(T) \leq mn_c d \sum_{t=1}^{T} [t^{-c_c\beta_j\delta'^2\theta_j/3}(1 - t^{-c_c\beta_j\delta_0^2/3}) + t^{-c_c\beta_j\delta_0^2/3}]. \tag{6.41}
$$

Let c_c be any value such that $c_c > [\eta\delta'^2/3]^{-1}$, then the regret of exploitation converges to a constant value as the following,

$$
R_I(T) \leq mn_c d(1 + \frac{3}{c_c\eta\delta'^2}). \tag{6.42}
$$

Thus the overall regret for semi-bandit feedback is

$$
\begin{aligned}
R(T) &= R_A(T) + R_I(T) \\
&\leq n_c c_c \log(T) + m n_c d(1 + \frac{1}{c_c \beta_j \delta'^2 \theta_j / 3 - 1}) \\
&= O(n_c d^2 \log(T)).
\end{aligned}
$$

□

Table 6.1: Comparison of regret bounds

d	$R(T)$ of LPR-BF		$R(T)$ of LPR-SF
	upper bound	lower bound	upper bound
$O(1)$	$O(n^3 \log T)$	$\Omega(n \log T)$	$O(n \log T)$
$O(\log n)$	$O(n^{3+\log(\frac{1}{\theta_{min}})} \log T)$	$\Omega(n \log T)$	$O(n^{1+\log(\frac{1}{\theta_{min}})} (\log n)^2 \log T)$
$O(n)$	$O((\frac{1}{\theta_{min}})^n n^3 \log T)$	$\Omega(n \log T)$	$O((\frac{1}{\theta_{min}})^n n^3 \log T)$

6.4.5 Discussion

Now we can compare the regret upper bound under the two observation models. Under bandit feedback, Algorithm LPR-BF gives $O((\frac{1}{\theta_{min}})^d n_b^3 \log(T))$. Under semi-bandit feedback model, Algorithm LPR-SF gives $O((\frac{1}{\theta_{min}})^d n_c d^2 \log(T))$. Both n_b and n_c are of order $O(n)$ for arbitrary topology. Comparison of the two upper bound is shown in Table 6.1. There is a positive gap in the regret upper bound between the semi-bandit feedback model and bandit feedback model, as long as $d = o(n_b)$.

6.5 Evaluation On Small Scale Network

We evaluate the performance of our algorithms in this section, with a packet level simulator we built.

6.5.1 Experiment Setup

To evaluate the performance of Algorithm LPR-BF and Algorithm LPR-SF, we build a simulator to run the algorithms. We use the RoofNet[2] data set and extract a

network topology with 38 nodes and 117 links. Picking two nodes from the topology and concatenating consecutive links without branching as one single logical link, we found the embedded graph between the source node and destination node as shown in Figure 6.1. In this subgraph, there are 7 nodes and 12 links. we construct 19

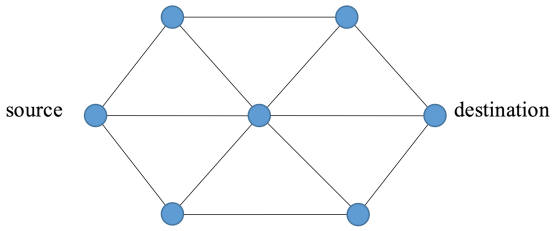


Figure 6.1: subgraph between the source and destination

paths from the source node to the destination node. Paths are allowed to revisit nodes but not links. The minimum size basis for these 19 paths contains 11 paths and the minimum link cover has 4 paths. Link success probabilities are generated at random following distribution *Uniform*(0.1,1). We generate ten different link success probability instances, each for 100 Monte Carlo runs, by default.

6.5.2 Benefit of Semi-Bandit Feedback

According to Theorems 26 and 28, the constants factors c_b and c_c needs to satisfy

$$c_b > 3\alpha_m^{-1} n_b^2,$$

and

$$c_c > 3\alpha_m^{-1} d^2,$$

where α_m is the reliability of the worst path, n_b is the size of the basis for bandit feedback model, and d is the maximum path length. The term α_m^{-1} yields a large values of c_b and c_c. We conjecture this term is not needed and can be replaced by one even though we cannot prove in theory. In the experiment though, we have the

freedom to explore how values of c_b and c_c affect the performance of our algorithm when we set

$$c_b = 3n_b^2, c_c = 3d^2.$$

In out setting, the constants are $c_b = 3n_b^2 = 363$ and $c_c = 3d^2 = 147$ (with factor α_m^{-1} removed from the equation). Hence each path in the basis is probed approximately $363 \log t$ times in Algorithm LPR-BF, and each path in the minimum cover is probed approximately $147 \log t$ times in Algorithm LPR-SF. Figures 6.2,6.3 show the average reward and regret of Algorithm LPR-BF and Algorithm LPR-SF over time of ten link success probability instances each with 100 simulation runs. Algorithm LPR-BF significantly out-performs Algorithm LPR-SF. Meanwhile, both algorithms identify the best path, as we see a steady, linear-like growth in the reward curves, and a very slow growth in the regret curves.

Observe in both figures, there is a ramp at the beginning of each curve. The cause of the ramp is that at the beginning of both algorithms, all the probes are used to explore the basis/cover paths in round-robin manner. Figure 6.4 show the behavior of the total number of probes for exploration, A_t, for the algorithms. Recall that at each time step t, if $A_t < n_{b,c}c_{b,c} \log t$, we increase A_t by one. Thus $A_t \approx \min\{t, nc \log t\}$, where we omit the subscripts for n and c. As shown in Figure 6.4, before A_t increases to the point where $t = nc \log t$, the value of A_t increases by one on every time step and all the probes go into exploration. Subsequently explorations and exploitations are interleaved. The slope of the ramps of the average reward curves are the average path success probability of the paths in the basis and the paths in the cover respectively. The length of the ramps, however, are determined by variables n and c. The transition point happens approximately at time t^*, s.t. $t^* = nc \log t^*$.

6.5.3 Size of basis for Bandit Feedback

In the case there are large number of paths and the size of the minimum basis of paths is small, it makes sense to explore only the basis and then use information of the basis to estimate the reliability of the rest of paths. However with our network, the size of basis is 11 while there are totally 19 paths. It's then worth checking whether the benefit of using a basis disappear on such small networks. We run our Algorithm LPR-BF under the case that the basis is composed of all the paths, and compare the performance with the case that we use a minimum-sized basis with 11 paths. Let n_b' and c_b' be the values of n_b and c_b we use for the case that all the paths forms the basis. Note that in this case in Algorithm LPR-BF, $n_b = 19$ and we don't have a lower bound on c_b anymore. For a convenient comparison, we select $c_b' = \frac{n_b c_b}{n_b'} = \frac{11}{19} c_b$, so that the two cases will have same length of the initial purely-exploration phase. Figure 6.5,6.6 compare the reward over time and regret over time of the two cases respectively. Averaged over 10 different link success probability instance and 100 Monte Carlo runs, the performance of using 11 paths as the basis is slightly better than that of using all 19 paths as the basis on the initial ramp phase, then the difference of the two remains the same as both have found the best path. The small difference on the slopes of the ramps in the curves of reward comes from the difference in average path success probability of the basis with 11 paths and that of the basis using all paths. In our case, the basis with 11 paths has a slightly larger average path success probability. Furthermore, this small benefit may not remain for a different selection of 11 paths as the basis, or with more link success probability instances, or if the underlying network topology changes.

6.5.4 Tuning constant factor $c_{b,c}$

Next we consider the effect of reducing c_b and c_c at the expense of not having a theoretical bound. By reducing c_b and c_c we reduce the length of the initial phase

of our algorithms where all the probes are used for exploration. It's preferable in practice if we can use a smaller number of probes and achieve the same reward, which corresponding to using smaller values for c_b(or c'_b) and c_c. However, the value of $c_{b,c}$ determine the number of samples each path in the basis/cover get during explorations of the algorithms. Basically, each path in the basis or cover is sampled approximately $c_{b,c} \log t$ times. Intuitively, if $c_{b,c}$ is too small, we don't get enough sample of path reliability in the basis or cover and hence there will be large estimation errors that prevent our algorithm to successfully identify the best path. To verify that, we run a set of simulations where we keep the underlying link success probability (and thus path success probability) fixed, and let c_b and c_c each vary over a set of values. We observe that the algorithms start to deviate from finding the best paths, as we decrease value of $c_{b,c}$. Figure 6.7, 6.8, 6.9 summarizes the mean and the 95% of regret over 100 Monte Carlo runs. The black curves show the mean value and the shades show the 95 percentiles of regret. For bandit feedback model with a basis of 11 paths, the regret deviates for $c_b = 200, 100, 50, 10$. The performance is especially bad when $c_b = 10$. The bandit feedback model using all 19 paths as the basis, however, behaves rather robustly against varying value of c'_b. It always find the best path with little deviation. As for the semi-bandit feedback model, the algorithm behaves well before c_c decreases to 20.

6.6 Evaluation on complete graphs

Here we evaluate LPR-BF and LPR-SF on a complete graph, where the total number of paths can be much larger than both the size of the basis and and the cover. We consider a complete graph with seven nodes, of which we remove the link that connects the source node and the destination node directly. Between the source and the destination, there are totally 325 paths. While the minimum size of a basis is 19, we investigate performance of LPR-BF with the minimum size basis and with

a basis that uses all 325 paths. For LPR-SF, we investigate a cover that contains all 3-hop paths. In the case of a seven-node clique, there are 20 paths in the cover. Link success probabilities are randomly generated from distribution *Uniform*(0.5,1). All simulations are repeated for one hundred times.

Does it make a difference whether the best path is inside or outside the basis? To answer this question, we fix the link success probabilities and generate ten bases that contain the best path and ten bases that do not. We run LPR-BF on these bases, 50 Monte Carlo runs for each, and compare the regret. The result show that bases containing the best path have slightly better performance than bases that do not contain the best path. This suggests that the benefit of having the best path inside the basis is minimal and it's not necessary to specifically identify a basis containing the best path for Algorithm LPR-BF. With this understanding, we proceed with the rest of the evaluation using bases that contain the best path.

Compared to the previous topology which has 11 paths in the minimum size basis, the seven-node clique network examined here has 19 paths in its minimum basis. If we choose c_b and c_c as in the previous section, then

$$c_b = 3n_b^2, c_c = 3d^2,$$

where $n_b = 19$ and $d = 6$. This produces a much larger c_b value and a comparable c_c value compared to the previous network. This considerably longthens the initial exploration phase for LPR-BF, as we observe in Figure 6.10. Meanwhile, c_c grows on the order of the square of the maximum path length. In this case it's six, the same as for the previous network. Hence the initial exploration phase for LPR-SF does not increase in length, and we observe that LPR-SF is able to quickly switch to the exploitation phase and correctly find the best paths.

For the same practical concern that it is desirable to shorten the initial exploration phase to speed up the algorithm, we speed up LPR-BF by using a smaller value of c_b.

Figures 6.11,6.12,6.13 show the regret for LPR-BF when we reduce constant c_b. The shaded areas show the 99 percentile and 95 percentile of regret of LPR-BF. When we scale c_b down by a factor of 0.4, LPR-BF still performs robustly to find the best path as the shaded area in Figures 6.11,6.12 show that in most of the Monte Carlo runs the regret grows slowly after the initial exploration phase. When the scale factor is lowered by 0.2, there are two runs during which LPR-BF does not find the correct best path, as shown by the light blue area in Figure 6.13. However for most of the runs Algorithms LPR-BF finds the best path, as shown with the dark blue area in Figure 6.13, and the regret remains low after the initial exploration phase. This suggests that we can scale down c_b with factor 0.2 in LPR-BF for the seven-node clique network.

We modify LPR-BF by scaling down constant c_b by a factor of 0.2, and compare its performance with some baseline algorithms, as shown in Figure 6.14. The algorithms we compare include LPR-BF with modified c_b (by a factor of 0.2) using minimum basis and the same algorithm using all paths as a basis, LPR-SF, and UCB1 algorithm[26]. The modified LPR-BF using a small basis out-performs UCB1, it also is much better than LPR-BF that uses a much larger basis. Algorithm LPR-SF performs the best among all 4 methods, with constant c_c unmodified.

6.7 Conclusions

In this chapter of the thesis, we design and analyze two online routing algorithms with different observation model: bandit feedback model, and semi-bandit feedback model. By developing the performance bounds for both algorithms, we show the potential benefit of having additional feedback information in the semi-bandit feedback model. Out theory gives guidelines of the performance when the problem scale is large. We simulation results evaluates the performance of our algorithms and show the benefit of exploring only the basis/cover of the paths.

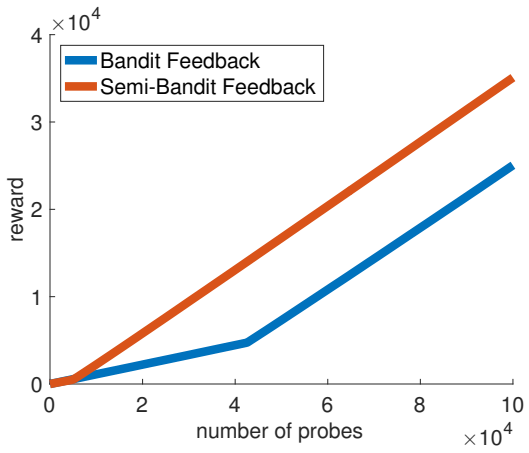


Figure 6.2: Average reward versus time (number of probes) of bandit feedback model and semi-bandit feedback model

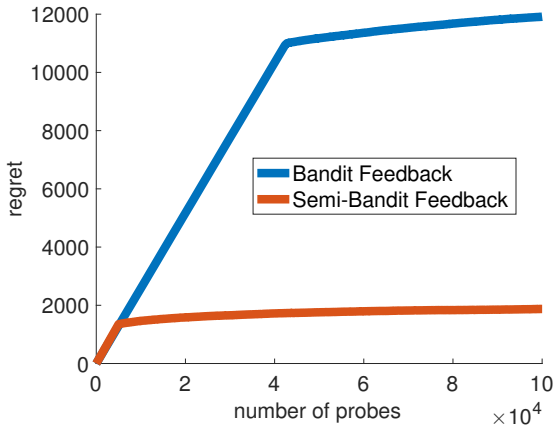


Figure 6.3: Regret versus time (number of probes) of bandit feedback model and semi-bandit feedback model

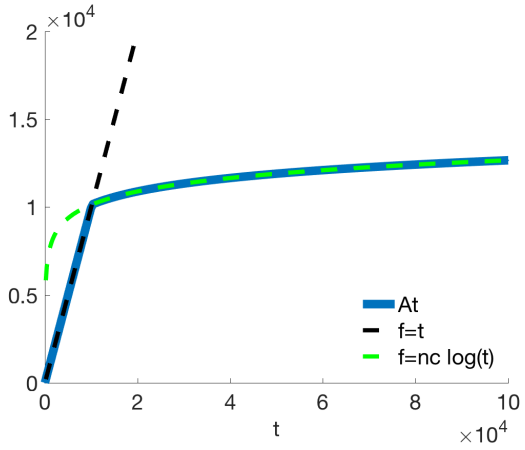


Figure 6.4: $At = \min\{t, nc \log t\}$, where $n = 11, c = 100$.

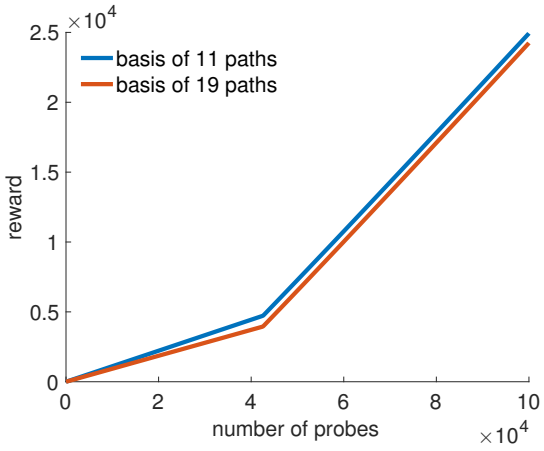


Figure 6.5: Average reward versus time (number of probes) of basis using 11 paths and basis using all 19 paths

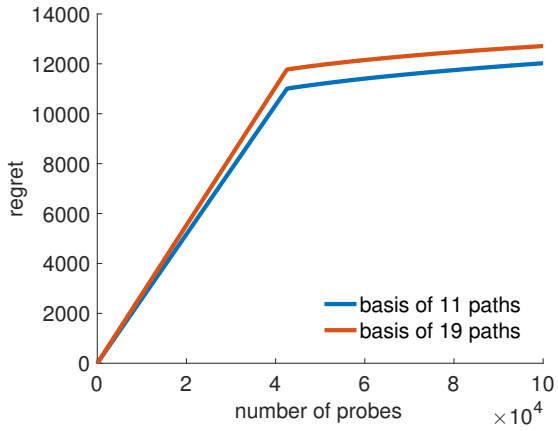


Figure 6.6: Average reward versus time (number of probes) of basis using 11 paths and basis using all 19 paths

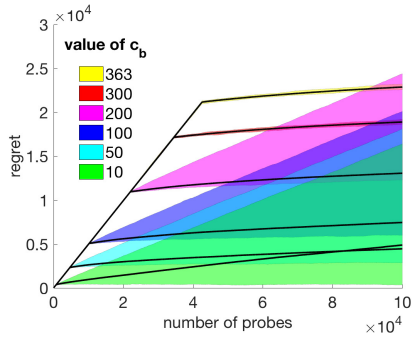


Figure 6.7: Regret and 95% for bandit feedback using a basis of 11 paths, where constant $c_b = \{363, 300, 200, 100, 50, 10\}$.

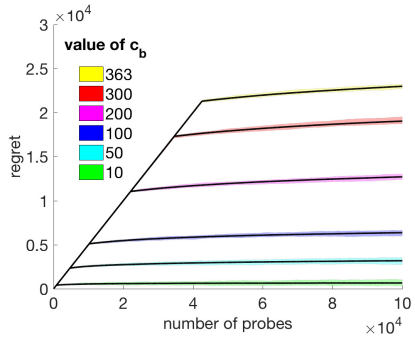


Figure 6.8: Regret and 95% for bandit feedback using all 19 paths as basis, where constant $c_b = \{363, 300, 200, 100, 50, 10\}$, and $c'_b = \frac{11}{19} c_b$.

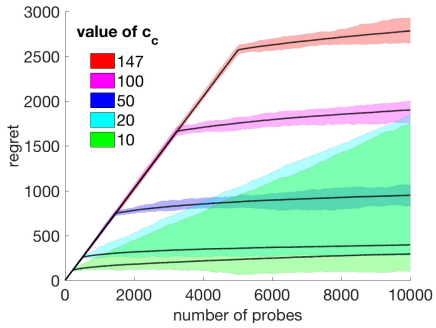


Figure 6.9: Regret and 95% for semi-bandit feedback using a cover of 4 paths, where constant $c_c = \{147, 100, 50, 20, 10\}$.

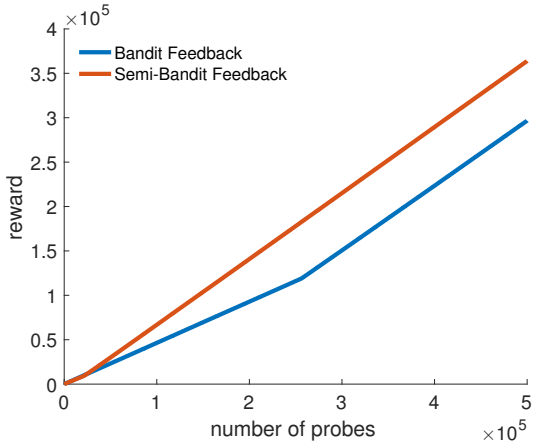


Figure 6.10: LPR-SF outperforms LPR-BF on the seven-node clique.

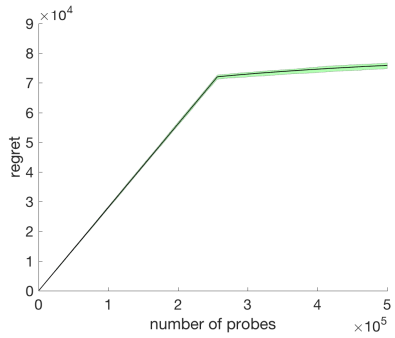


Figure 6.11: Regret and 99% for LPR-BF using a basis of 19 paths, where $c_b = 3n_b^2 = 1083$.

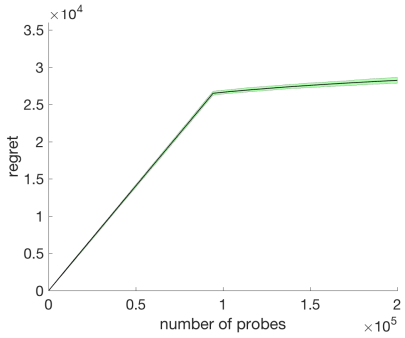


Figure 6.12: Regret and 99% for LPR-BF using a basis of 19 paths, where c_b is scaled down by a factor 0.4, $c_b = 3n_b \times 0.4 = 433$.

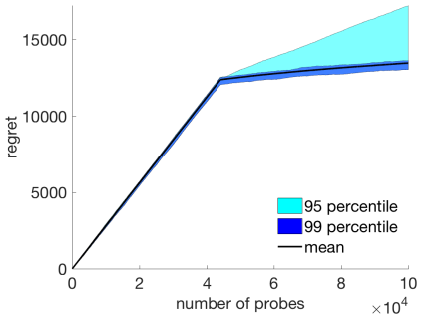


Figure 6.13: Regret and 95%, 99% for LPR-BF using a basis of 19 paths, where c_b is scaled down by a factor 0.2, $c_b = 3n_b \times 0.2 = 216$.

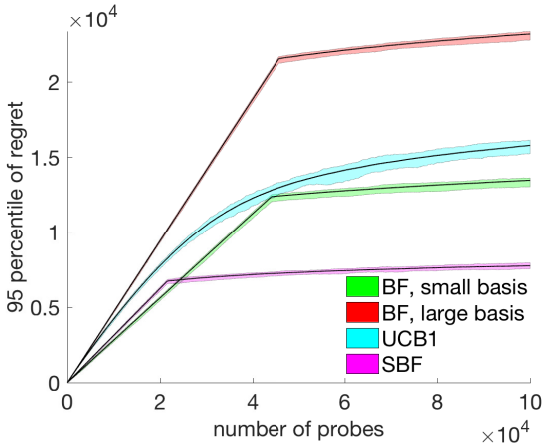


Figure 6.14: 95% of the regret for LPR-BF with small basis (19 paths), LPR-BF large basis (all 325 paths in the network), LPR-SF, and UCB1.

CHAPTER 7

CONCLUSIONS AND REMAINING WORK

7.1 Conclusions

This thesis has investigated and explored how to design inference techniques for application in large scale networks and how to improve inference efficiency with designed measurements.

The thesis first looked at the server selection problem in web content delivery. We proposed a randomized algorithm that firstly let each user randomly selection a small set of servers, and then based on our proposed method that infers congestions status at the servers we proposed to use the Go-With-The-Winner approach and let users select the server that gives the good performance (indicating no congestion) and drop all the other servers.

We then investigated measurement design for network tomography. In the problem to infer link metric through path level measurements in a network, we quantify the information each path level measurement contains by Fisher Information Matrix (FIM). We show that by maximizing optimal criteria of the FIM we can minimizing the estimation error of the link metrics and thus achieve optimal allocations of measurements among paths.

Following that, we further compared the performance of loss tomography using multicast measurements with loss tomography using unicast measurements, in tree topologies.

In the final part of the thesis, we investigated the design of online routing with regard to path reliability, where the goal is to adaptively find the most reliable path and to minimize the cost caused by transmission losses during this process.

7.2 Future work

For sever selection in Content Delivery Networks, it's interesting to further investigate the possible solution of having a subset of the users run our Go-With-The-Winner algorithm while the other users each choose a single server uniformly at random. It's also worth further analyzing and evaluating that how the content popularity distribution affects the performance of our algorithm.

For the problem of experiment design for network tomography, there is space on designing general probe allocation solution for the case that there are more paths than number of links. Meanwhile, it's worth noticing that even though the closed-form solution we developed is for solving probe allocation for the network tomography problem, yet our problem formulation is more general and thus our solution can potentially be extended to other different applications that fits the formulation. It's worth finding out other problems or applications our solution can be applied to.

As to our work of comparing multicast and unicast measurements for link loss tomography, our current results are mostly empirical evaluation. It will another contribution if we can provide more results on the comparison of Fisher Information and performance bound.

Last but not least, for the problem of learning path reliability for online routing. One possible extension is to improve the current proof technique and explore if tighter bounds are possible for the two observation models. It will also be nice if performance of the algorithms can be evaluated on larger networks.

BIBLIOGRAPHY

[1] Adhikari, Vijay Kumar, Guo, Yang, Hao, Fang, Varvello, Matteo, Hilt, Volker, Steiner, Moritz, and Zhang, Zhi-Li. Unreeling netflix: Understanding and improving multi-cdn movie delivery. In *INFOCOM, 2012 Proceedings IEEE* (2012), IEEE, pp. 1620–1628.

[2] Aguayo, Daniel, Bicket, John, Biswas, Sanjit, Judd, Glenn, and Morris, Robert. Link-level measurements from an 802.11b mesh network. In *SIGCOMM* (2004).

[3] Akamai. Akamai download manager. `http://www.akamai.com/html/solutions/downloadmanager_overview.html`.

[4] Atkinson, A.C., and Donev, A.N. *Optimum Experimental Designs*. Clarendon Press, 1992.

[5] Bellare, Mihir, and Kohno, Tadayoshi. Hash function balance and its impact on birthday attacks. In *Advances in Cryptology-Eurocrypt 2004* (2004), Springer.

[6] Bhorkar, Abhijeet A., Naghshvar, Mohammad, Javidi, Tara, and Rao, Bhaskar D. Adaptive opportunistic routing for wireless ad hoc networks. *IEEE/ACM Trans. Netw. 20*, 1 (Feb. 2012), 243–256.

[7] Boyd, S., and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, 2004.

[8] Breslau, Lee, Cue, Pei, Cao, Pei, Fan, Li, Phillips, Graham, and Shenker, Scott. Web caching and zipf-like distributions: Evidence and implications. In *INFOCOM* (1999).

[9] Bu, T., Duffield, N., Lo Presti, F., and Towsley, D. Network tomography on general topologies. In *SIGMETRICS* (2002), ACM.

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

[11] Caceres, R., Duffield, N.G., Horowitz, J., Towsley, D., and Bu, T. Multicast-based inference of network-internal characteristics: Accuracy of packet loss estimation. *IEEE Trans. Info. Thy 45* (1998), 2462–2480.

[12] Casella, G., and Berger, R.L. *Statistical Inference*. Duxbury advanced series. Duxbury Thomson Learning, 2002.

[13] Coates, M., Hero, A. O., Nowak, R., and Yu, B. Internet tomography. *IEEE Signal Processing Magzine 19* (2002), 47–65.

[14] Cole, Richard, Maggs, Bruce M, Mitzenmacher, Michael, Richa, Andréa W, Schröder, Klaus, Sitaraman, Ramesh K, Vöcking, Berthold, et al. Randomized protocols for low-congestion circuit routing in multistage interconnection networks. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing* (1998), ACM, pp. 378–388.

[15] Crovella, Mark E, and Carter, Robert L. Dynamic server selection in the internet. Tech. rep., Boston University Computer Science Department, 1995.

[16] Dilley, John, Maggs, Bruce, Parikh, Jay, Prokop, Harald, Sitaraman, Ramesh, and Weihl, Bill. Globally distributed content delivery. *Internet Computing, IEEE 6*, 5 (2002), 50–58.

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

[18] Duffield, N.G., and Lo Presti, F. Multicast inference of packet delay variance at interior network links. In *IEEE INFOCOM* (2000).

[19] Dykes, Sandra G, Robbins, Kay A, and Jeffery, Clinton L. An empirical evaluation of client-side server selection algorithms. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE* (2000), vol. 3, IEEE, pp. 1361–1370.

[20] Fricker, Christine, Robert, Philippe, and Roberts, James. A versatile and accurate approximation for lru cache performance. In *Proceedings of the 24th International Teletraffic Congress*, ITC '12.

[21] Gai, Yi, Krishnamachari, Bhaskar, and Jain, Rahul. Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations. *IEEE/ACM Trans. Netw. 20*, 5 (Oct. 2012), 1466–1478.

[22] Gill, Phillipa, Arlitt, Martin, Li, Zongpeng, and Mahanti, Anirban. Youtube traffic characterization: A view from the edge. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement* (2007), IMC '07, ACM.

[23] Gu, Yu, Jiang, Guofei, Singh, Vishal, and Zhang, Yueping. Optimal probing for unicast network delay tomography. In *IEEE INFOCOM* (2010).

[24] He, Ting, Goeckel, Dennis, Raghavendra, Ramya, and Towsley, Don. Endhost-based shortest path routing in dynamic networks: An online learning approach. In *INFOCOM, 2013 Proceedings IEEE* (2013), IEEE, pp. 2202–2210.

[25] He, Ting, Liu, Chang, Swami, Ananthram, Towsley, Don, Salonidis, Theodoros, Bejan, Andrei Iu., and Yu, Paul. Fisher information-based experiment design for network tomography. In *SIGMETRICS* (2015), ACM.

[26] Kaufmann, Emilie, Cappé, Olivier, and Garivier, Aurélien. On bayesian upper confidence bounds for bandit problems. In *AISTATS* (2012), pp. 592–600.

[27] Key, P., Massoulie, L., and Towsley, P.D. Path selection and multipath congestion control. *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE* (2007).

[28] Krishnamurthy, A., and Singh, A. Robust multi-source network tomography using selective probes. In *IEEE INFOCOM* (2012).

[29] Krishnan, S Shunmuga, and Sitaraman, Ramesh K. Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. In *Proceedings of the 2012 ACM conference on Internet measurement conference* (2012), ACM, pp. 211–224.

[30] Lai, T.L, and Robbins, Herbert. Asymptotically efficient adaptive allocation rules. *Adv. Appl. Math. 6*, 1 (Mar. 1985), 4–22.

[31] Liu, Cricket, and Albitz, Paul. *DNS and Bind*. O'Reilly Media, Inc., 2009.

[32] Lo Presti, F., Duffield, N.G., Horowitz, J., and Towsley, D. Multicast-based inference of network-internal delay distributions. *IEEE/ACM Trans. Networking 10*, 6 (Dec. 2002), 761–775.

[33] Ma, Liang, He, Ting, Leung, Kin K., Swami, Ananthram, and Towsley, Don. Identifiability of link metrics based on end-to-end path measurements. In *ACM IMC* (2013).

[34] Ma, Liang, He, Ting, Leung, Kin K., Towsley, Don, and Swami, Ananthram. Efficient identification of additive link metrics via network tomography. In *ICDCS* (2013).

[35] Mitzenmacher, Michael, Richa, Andréa W, and Sitaraman, Ramesh. The power of two random choices: A survey of techniques and results. *COMBINATORIAL OPTIMIZATION-DORDRECHT- 9*, 1 (2001), 255–304.

[36] Mitzenmacher, Michael, and Upfal, Eli. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, New York, NY, USA, 2005.

[37] Mitzenmacher, Michael, and Upfal, Eli. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, New York, NY, USA, 2005.

[38] Nguyen, Hung Xuan, Figueiredo, Daniel R., Grossglauser, Matthias, and Thiran, Patrick. Balanced relay allocation on heterogeneous unstructured overlays. In *INFOCOM* (2008).

[39] Nygren, Erik, Sitaraman, Ramesh K, and Sun, Jennifer. The akamai network: a platform for high-performance internet applications. *ACM SIGOPS Operating Systems Review 44*, 3 (2010), 2–19.

[40] Ousterhout, Kay, Wendell, Patrick, Zaharia, Matei, and Stoica, Ion. Sparrow: Distributed, low latency scheduling. In *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, ACM.

[41] Poor, H. Vincent. An introduction to signal detection and estimation. Springer, 1994.

[42] Raab, Martin, and Steger, Angelika. balls into binsa simple and tight analysis. In *Randomization and Approximation Techniques in Computer Science*. Springer, 1998, pp. 159–170.

[43] Ramasubramanian, V., Malkhi, D., Kuhn, F., Balakrishnan, M., Gupta, A., and Akella, A. On the treeness of internet latency and bandwidth. In *SIGMETRICS* (2009), ACM.

[44] Rothenberg, T. J. Identification in parametric models. *Econometrica 39* (May 1971), 577–591.

[45] Sayal, Mehmet, Breitbart, Yuri, Scheuermann, Peter, and Vingralek, Radek. Selection algorithms for replicated web servers. *ACM SIGMETRICS Performance Evaluation Review 26*, 3 (1998), 44–50.

[46] Shih, Meng-Fu, and Hero, A. Unicast inference of network link delay distributions from edge measurements. In *IEEE ICASSP* (2001).

[47] Singhal, Harsh, and Michailidis, George. Optimal sampling in state space models with applications to network monitoring. In *ACM SIGMETRICS* (2008).

[48] Torres, Ruben, Finamore, Alessandro, Kim, Jin Ryong, Mellia, Marco, Munafo, Maurizio M, and Rao, Sanjay. Dissecting video server selection strategies in the youtube cdn. In *Distributed Computing Systems (ICDCS), 2011 31st International Conference on* (2011), IEEE, pp. 248–257.

[49] Trees, H. L. Van. *Detection, estimation, and modulation theory.* John Wiley&Sons, 2004.

[50] Wendell, Patrick, Jiang, Joe Wenjie, Freedman, Michael J, and Rexford, Jennifer. Donar: decentralized server selection for cloud services. In *ACM SIGCOMM Computer Communication Review* (2010), vol. 40, ACM.

[51] Xi, Bowei, Michailidis, George, and Nair, Vijayan N. Estimating network loss rates using active tomography. *Journal of the American Statistical Association 101*, 476 (2006), 1430–1448.

[52] Ying, Lei, Srikant, R, and Kang, Xiaohan. The power of slightly more than one sample in randomized load balancing. In *Computer Communications (INFOCOM), 2015 IEEE Conference on* (2015), IEEE.

[53] Zhao, Yao, Chen, Yan, and Bindel, David. Towards unbiased end-to-end network diagnosis. In *SIGCOMM* (2006).