Cognitive Optical Network Architecture in Dynamic Environments

by

Xijia Zheng

E.E., Massachusetts Institute of Technology (2017)
S.M., Massachusetts Institute of Technology (2015)
B.Eng., The University of Hong Kong (2014)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2020

Author .....
Department of Electrical Engineering and Computer Science
May 15, 2020

Certified by .....
Vincent W. S. Chan
Joan and Irwin Jacobs Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Cognitive Optical Network Architecture in Dynamic Environments

by

Xijia Zheng

Submitted to the Department of Electrical Engineering and Computer Science
on May 15, 2020, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering

Abstract

Emerging network traffic requires a more agile network management and control system to deal with the dynamic network environments than today's networks use. The bursty and large data transactions introduced by new technological applications can cause both high costs and extreme congestion in networks. The prohibitive cost of massive over-provisioning will manifest as huge congestions during peak demand periods. The network management and control system must be able to sense the traffic changes and reconfigure in a timely manner (in tens of milliseconds instead of minutes or hours) to use network resources efficiently. We propose the use of cognitive techniques for fast and adaptive network management and control of future optical networks. The goal of this work is to provide timely network reconfigurations in response to dynamic traffic environments and prevent congestion from building up.

We make a simplified model of the expected traffic arrival rate changes as a multi-state Markov process based on the characteristics of the dynamic, bursty, and high granularity traffic. The traffic is categorized into different network traffic environments by the length of the network coherence time, which is the time that the traffic is unvarying. The tunneled network architecture is adopted due to its supremacy in reducing the control complexity when the traffic volume is at least one wavelength.

In the long coherence time regime where traffic changes very slowly, the traffic detection performances of two Bayesian estimators and a stopping-trial (sequential) estimator are examined, based on the transient behaviors of networks. The stopping-trial estimator has the fastest response time to the changes of traffic arrival statistics. We propose a wavelength reconfiguration algorithm with continuous assessment where the system reconfigures whenever it deems necessary. The reconfiguration can involve addition or subtraction of multiple wavelengths. Using the fastest detection and reconfiguration algorithm can reduce queueing delays during traffic surges without over-provisioning and thus can reduce network capital expenditure and prevent wasting resources on erroneous decisions when surges occur.

For traffic with moderate coherence time (where traffic changes at a moderate rate) and the short coherence time (where traffic changes quickly), the stopping-

trial estimator still responds to the traffic changes with a short detection time. As long as the inter-arrival times of traffic transactions are independent, the algorithm is still optimum. The algorithm provides no prejudice on the exact network traffic distribution, avoiding having to sense and estimate detailed arrival traffic statistics.

To deal with fast-changing traffic, we model the transient convergent behaviors of network traffic drift as a result of traffic transition rate changes and validate the feasibility and utility of the traffic prediction. In a simple example when the network traffic rate changes monotonically in a linear model, the sequential maximum likelihood estimator will capture the traffic trend with a small number of arrivals. The traffic trend prediction can help to provide fast reconfiguration, which is very important for maintaining quality of service during large traffic shifts.

We further investigate the design of an efficient rerouting algorithm to maintain users' quality of service when the incremental traffic cannot be accommodated on the primary path. The algorithm includes the fast reconfiguration of wavelengths in the existing lit and spatially routed fibers, and the setting up and lighting of new fibers. Rerouting is necessary to maintain users' quality of service when the queueing delay on the primary path (determined by shortest path routing) exceeds the requirement. Our algorithm triggers reconfiguration when a queueing delay threshold is crossed on the primary path. The triggering by a threshold on the queueing delay is used due to its simplicity, and it is directly measurable by the exact traffic transaction sizes and the queue size, which reflect both the current network traffic environment and the network configurations. A dynamic rerouting algorithm implemented with a shortest-path algorithm is proposed to find the secondary paths for rerouting. We make the conjecture that it is desirable that the alternate paths for rerouting have small numbers of hops and are disjoint with other busy paths when the hops on the path are independent. In addition, the conjecture suggests that a good candidate network topology should have high edge-connectivity. Wavelength reservation for rerouted traffic does not maximize wavelength utilization. We make the conjecture that traffic with different sizes should be broken up into multi-classes with dedicated partitioned resources and the queueing delay should be normalized by the transmission time for rerouting triggering to realize better network utilization.

Thesis Supervisor: Vincent W. S. Chan
Title: Joan and Irwin Jacobs Professor of Electrical Engineering and Computer Science

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my research advisor, Professor Vincent W. S. Chan, for his knowledge, expertise, support, and patience. His emphasis on practical design, creative thinking and precise communication continues to be a great inspiration to me. Moreover, his dedication and advice have been invaluable and will continue to support me in life.

I would also like to thank Dr. Richard Barry and Professor Vivienne Sze for carving out time to join my thesis committee. I am deeply grateful to them for providing invaluable feedback for my research and assisting in polishing this work.

I would like to thank my fellow group mates: Lei Zhang, Henna Huang, Matthew Carey, Manishika Agaskar, Shane Fink, Arman Rezaee, John Metzger, Antonia Feffer, Esther Jang, Andrew Song, and Junior Weerachai Neeranartvong. All of them have made my time in the group enjoyable and wonderful.

I would like to thank my friends both near and far. I am so grateful for their friendship and encouragement.

Last but not the least, I would like to thank my parents for their unconditional love, guidance, and unwavering support. This thesis is dedicated to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The dynamic large traffic generated by diverse technology applications will drive the architecture of future optical networks. The bursty, unscheduled, and large data transactions introduced by new technological applications can cause both high costs and extreme congestions in networks. In the current quasi-statically configured network, lack of over-provisioning will manifest as huge congestions during peak demand periods. The dynamic and bursty (unpredictable) nature of large traffic transactions either requires over-provisioning of the networks, which is costly, or a more agile network control and management system that adaptively allocates resources by reconfiguring the network in a timely manner in reaction to the offered traffic. The network management and control system should be able to sense traffic changes and make reconfigurations to use network resources efficiently. Specifically, reconfigurations should be done on a subsecond time scale with no human involvement. To meet these demands, cognitive networking is proposed as a candidate architectural construct that can provide fast, dynamic, and efficient control using cognitive techniques.

1.1 Increasing Traffic

Emerging network applications will make the offered traffic in future optical networks more bursty with high granularity sessions from megabytes to terabytes. The high

granularity indicates the increased volume of large traffic. Compared to the past when major traffic is only in the range of bits to megabytes, now much larger traffic occurs in gigabytes or even in terabytes. Within five to ten years, current optical networks are predicted to have an approximately three to four orders of magnitude increase in data rates, mostly due to large transactions. It is reported in [16] that IP video traffic will be 82% of all IP traffic (both business and consumer) by 2022 globally. As indicated in [15], an Internet-enabled High-Definition (HD) television that draws two to three hours of content per day from the Internet would generate a similar amount of traffic on average as the Internet traffic generated by an entire household today. Moreover, the increasing demands of Ultra-High-Definition (UHD) with 4K or even 8K video streaming will have more profound effects on the traffic growth due to the increase in the bit rate. The bit rate for 4K video is about 15 to 18 Mbps, which is more than double the HD video bit rate and nine times more than the bit rate of Standard-Definition (SD) video [15]. It is estimated in [15] that two-thirds of the installed flat-panel TV sets will be UHD by 2023, up from 33% in 2018. Besides, various new devices and new applications make the traffic patterns more diverse. Among them, Machine-to-Machine (M2M) applications are the major contributors, such as video surveillance, healthcare monitoring, smart meters, transportation, package or asset tracking, etc [15]. It is predicted in [15] that M2M connections will be half of the total devices and connections by 2023. All the dynamic large traffic generated by the diversity of technology applications will drive the architecture of future optical networks.

The extreme burstiness and high granularity of the network traffic is challenging today's optical networks, as it requires optical network control to be much more dynamic unless expensive large over-provisioning is used. Due to the dynamic traffic, statistical multiplexing to smooth quiescent flows between major nodes will not occur most of the time. The highly complex and dynamic networking environments will pose serious challenges for physical and higher network layers architecture, and greatly increase the network operating costs. The current network management and control systems of the backbone optical networking is much too slow to handle the variety

of traffic that comes from modern integrated global heterogenous networks, which include but are not limited to wireless networks, satellites networks, etc. It is because the current architectures of optical networks were designed for the quasi-static statistically multiplexed traffic of the past. Even for current software-defined-networks (SDN), network function virtualization (NFV), and orchestration, their network reconfiguration times range from tens of minutes to hours and even to days, which are inadequate to deal with current second-scale traffic changes [44]. The backbone optical networking needs to support different infrastructures from many access networks for millions of mobile users, and various traditional and new network services, such as data centers/private clouds [23], Software as a Service (SaaS), Infrastructure as a Service(IaaS), Internet-of-Things(IoT) [1], video-based searches [22], etc.

The increasing dynamic and high-granularity traffic will require a much smarter network management and control system to rapidly adapt to bursty applications and their service needs, especially for the increasingly large-volumed "elephants". Unpredictable dynamic traffic changes require quick network adaptation on a scale of seconds to maintain users' quality of service, such as the service delay. Moreover, service-based architecture requires dynamically tailored network configurations. In the near term (2020), we expect an active control plane in minutes for network orchestration [44], such as dynamic load balancing. In the mid term (2021+), a very active control plane is expected with functions including agile access points, per-session routing (segment routing), and the Internet of Things(IoT)/smart-city applications [1]. In the far term(2023+), we expect more dynamic network functions such as mid-session rerouting, etc. The control planes are designed for time critical services and network security and safety[12]. Network adaptions are expected to be performed on a scale of a second ($\sim$ 100 milliseconds - 1 second). Therefore, efforts should be made to develop a new optical network control and management scheme that can quickly sense the network and reconfigure within a much shorter amount of time than before, preferably as quickly as statistically viable. The new scheme should reduce human involvement, since humans in the loop of current networks contribute to the most reconfiguration time and control costs. Meanwhile, the network control and man-

agement system should be kept simple and efficient enough to reduce the burdens of network operations.

1.2 Cognitive Optical Networking

Cognitive networking is a desirable architecture construct that can provide fast, dynamic, and efficient control using automated cognitive techniques to meet the above requirements. As defined in [45][46], a cognitive network can sense current network conditions, and then autonomously learn and make decisions from these conditions to realize end-to-end goals. Its cognitive control and management scheme can respond to network changes and reconfigure the networks in a short amount of time without tedious network tuning by humans. For current network architectures, the intrinsic system complexity will increase explosively in the near future, but the current practice has a limit that makes such increase computationally intractable and expensive. When intelligence is introduced, the operational complexity will come down. In short, intelligence enables the adaptive network to scale. The goal of cognitive networking is ultimately fully automated networks without any human involvement [45][46].

The exploration of cognitive techniques on optical networks started only a decade ago. The idea of incorporating cognitive techniques into networks was originally introduced to wireless systems and networks to solve the spectrum sensing and allocation problem in the 2000s [45] [35], such as the idea of cognitive radios. The idea of using a large amount of information gained from the experience of network nodes to improve the overall network and user performance is proposed in [34]. Increasingly, the idea of cognitive networks is growing beyond the use of cognitive radios to improve the performance of heterogeneous networks. The exploration of the use of cognitive techniques in the domain of optical networks started around 2010. However, most researchers have focused on using cognitive techniques to solve specific optical network operating problems, and the design of comprehensive cognitive optical network control and management schemes are still conceptual [53, 52, 48, 36, 19, 20, 43, 42, 6].

Several groups of researchers have proposed the designs of general cognitive con-

trol and management schemes for optical networks, but these ideas are still far from practical implementation. Zervas and Simeonidou have briefly provided a prototypical distributed cognitive architecture called COGNITION [53, 52]. Cognition is expected to be implemented from the bottom physical layer to the top application layer of the network architecture across one or multiple domains. However, more implementation details and verifications are needed, especially on the complicated network management and control plane. Wei et al. have proposed a cognitive optical substrate with a mesh topology only in the core networks. This substrate aims to provide high-speed, bandwidth-on-demand and rapidly-adaptive wavelength services in a client-service-aware approach [48]. However, the design of the substrate concerns only the physical layer and has no coordination with higher-layer functionalities. The project Cognitive Heterogeneous Reconfigurable Optical Networks (CHRON) has been proposed by several groups of researchers in Europe [14] to improve the dynamicity of the optical network control planes. The project has investigated in the intelligent monitoring techniques, the cross-layer cognitive control architecture design, and the multi-objective optimization of the performance in term of cost and energy efficiency [14]. Monroy et al. have designed a network control plane architecture in CHRON [36, 19]. The control plane of CHRON includes a cognitive decision system and a network monitoring system. Later, de Miguel et al. have successively elaborated the centralized cognitive framework and built a testbed [20, 43, 42, 6]. However, their design has not comprehensively solved the efficient network monitoring problem or the fast reconfiguration problem. Moreover, the four-node network topology in their testbed is too simple to validate the performance of the design in real-life networks, which usually have hundreds of nodes. In particular, there is no work that shows that reconfiguration can be done based on observed traffic changes. In summary, a comprehensive cognitive network management and control scheme for current optical networks is needed.

Chan et al. have proposed a detailed cognitive all-optical network architecture in [10, 9, 12]. Figure 1-1 shows the concept of the cognitive all-optical network with optical gateways between hierarchical subnet, peering gateway control points, and

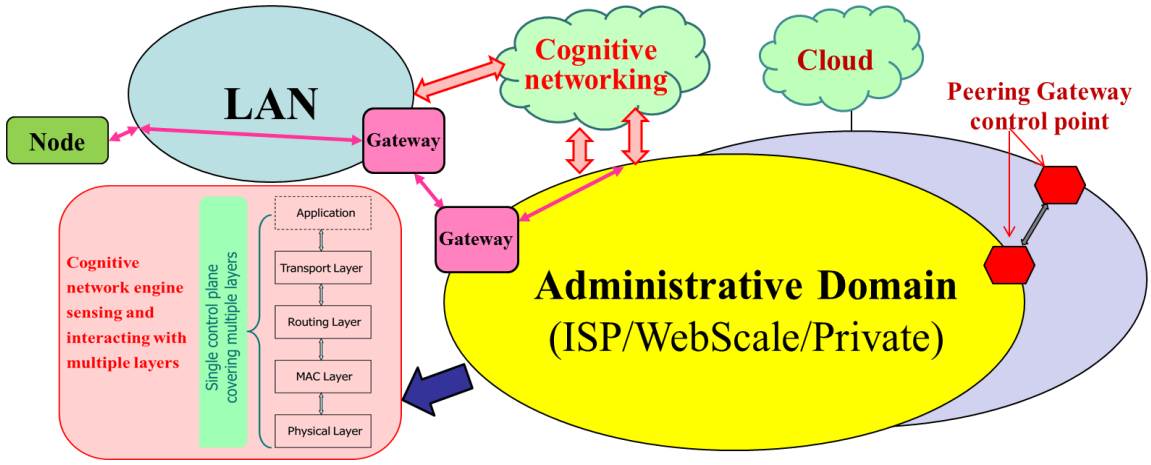

Figure 1-1: Cognitive all-optical network concept with optical gateways between hierarchical subnet, peering gateway control points and cognitive engine sensing and interacting with multiple layers for network control. Reproduced from [7, 9].

cognitive engine sensing and interacting with multiple layers for network control from the works in [7, 9]. The cognitive engine senses and interacts with multiple layers to perform the dynamic control of the whole network system, such as the scalable fault management with dynamic probing of the network states in [50], the fast scheduling algorithm with a probing approach to enable the setup of end-to-end connections [54]. It may reside at network nodes as well as at a centralized or distributed controller/s. Figure 1-2 further elaborates a desired development of agile, responsive, and affordable on-demand network services via new network management and control architecture across all network layers from Layer 1 Physical Layer to Layer 7 Application Layer [8] [10]. The cognitive network management and control architecture will incorporate with the upgraded physical layer to support big and bursty "elephant" traffic [55, 56]. The major technology advances in physical layer include the massive integration via silicon photonics and hybrids for reduction in footprints (increasing density) lower costs, weight, and power consumption [55]. A dynamic transport layer protocol and data center routing topology and partitioning of resources is required as shown in [26].

A cognitive network management and control system can sense the current network state conditions, such as traffic and flow patterns, and uses this information to decide how to adapt the network to satisfy or improve overall performance and provide quick responses to transaction requests [7, 9]. The cognitive network module is part of the control plane that touches all layers of a network. Traditionally, users never interact with the control plane [55, 56]. In the future optical network, users can interact via Application Layer or Transport Layer, and off-band control network. Interactions necessitated by sudden traffic changes of elephant traffic require fast adaptations.

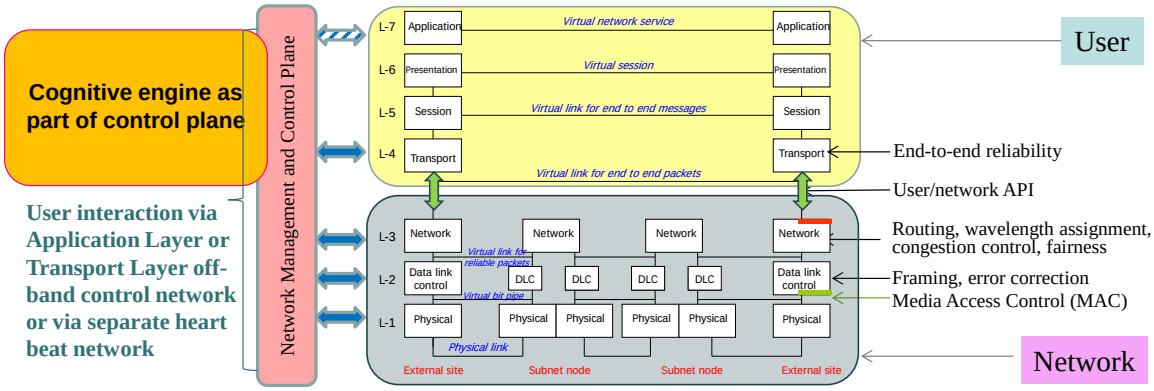

Figure 1-2: An agile control plane architecture of cognitive network management and control system. Reproduced from [8, 10].

In summary, the desired new network management and control architecture in cognitive optical networking should have the following features:

1. Inference of network states based on traffic and active probing often with sparse and stale data [50, 54];

2. All-optically switched architecture that provides agility and efficient resource usage [11, 55];

3. Decisions and actions taken on load balancing, reconfiguration, restoration [50, 55];

4. Cognitive techniques that predict optimum configuration for fast adaptation to improves network performance without detailed assumptions of channel and traffic statistics [31, 10].

The first point is discussed in [38, 40] with the design of the cognitive significant state sampling for a cost-effective network management system. It is shown in [38] that the adaptive monitoring system can greatly reduce the network management and control overhead if the network states information is only collected when it can improve network performance. The second point is discussed in [11, 49, 24, 27, 28, 55, 56, 57] with the comprehensive design of Optical Flow Switching (OFS). OFS is an agile all-optical end-to-end network service for users with large traffic flows that bypass routers. The dynamic per flow on a time scale of $1-10^4$ seconds scheduling can prevent collisions, and off-band signaling is used for reservation, scheduling, and setup on a time scale of < 100 milliseconds. The fourth point is mentioned in [31, 10], where the cognitive method for routing with the potential for improvements in network performance is discussed. In [31], the inference and estimation methods are used on the network traffic to modify the parameters of their routing protocols and/or routing tables to improve performance metrics such as packet delay or network throughput.

In this work, we will mainly focus on the third and fourth points as our design goals to enable cognitive optical networking control and management to be done in a timely manner. We will try to optimize network configurations to improve network performance based on the traffic information.

1.3 Design Goals

The goal of this work is to provide insights into the design of a comprehensive and practical cognitive control and management scheme for optical networks. Specifically, the design should provide an efficient and simple way to implement architectures that can avoid a big burden for network management and control, and to improve the cost efficiency of dynamic large transactions.

As a result, the control system should be able to perform rapid adaptations to

maintain a low queueing delay with the fastest possible algorithms (and that means no human in the loop). This is extremely important for the network in terms of stability, scalability, reliability, and evolvability. It not only guarantees that optical networks can function well in the traffic with great granularity, but it has to be scalable when the network grows larger and faster. With the fast dynamics noted, agile cognitive optical network management and control systems will guarantee both the quality of transmission and network robustness with little or no involvement of humans, and will finally move towards intelligent automatic control of networks. All of these make it more affordable to meet changing requirements and incorporate new technology as the network evolves.

Based on the design goal mentioned above, this work will mainly focus on two modules to enable cognitive optical networking control and management in a timely manner with little or no human involvement as:

1. **Traffic detection and estimation – fast recognition of the nonstationary traffic changes;**

2. **Network reconfiguration – quick adaption based on input quality of service metric (delay, cost, etc.) and optimum sequential decision algorithm.**

The traffic detection and estimation module focuses on an efficient way to detect and estimate the network traffic state information for sequential accurate network reconfiguration. Traditional approaches to traffic change detection and estimation require the collection of excessive historical network traffic information to allow for fast and agile reconfiguration of network resources. The prevalence of dynamic traffic sessions in today's networks, which manifest themselves through frequent and bursty changes of the network states, makes such approaches ineffective and even infeasible sometimes. Therefore, we need to design a traffic detection and estimation scheme that can make a decision in the shortest possible time when the traffic statistics provide enough confidence for reconfiguration to enhance the ability of the network to respond to changes efficiently.

The network reconfiguration module focuses on accurately adapting the network configurations based on the current operating conditions. In this work, three major reconfiguration options are used in order of preference for the reconfigurable network architecture are:

1. Lighting up a wavelength in the same primary lightpath between a node pair;
2. Rerouting incremental traffic to a secondary path with open wavelengths between the node pair;
3. Lighting up a new fiber with multiple wavelengths together with optical switching to accommodate overflowing traffic.

The reconfiguration can leverage the historical data as well as the real-time information acquired from monitoring to guide the decision process. Accurate prediction of future network states based on past traffic as well as active probing algorithms are essential for any modern management protocol.

1.4 Cognitive Optical Network Design Submodules

We further break down the two general modules in the previous section into the following submodules to guide the design of a fast-reconfigurable network. The core of cognition in networking is self-initiated actions. To be more specific, the networks are equipped with the capability to flexibly sense network conditions and then reconfigure dynamically. Also, network costs need to be considered to guarantee that architecture is affordable to be implemented widely. To complete this fast-reconfigurable and economical cognitive optical network architecture design, the following points of network management and control scheme needed to be stressed.

1. Define the dynamic network traffic environment. This section will focus on how to depict different network traffic environments.

2. Detect and estimate current network traffic fast and efficiently. This problem will focus on efficiently detecting current real-time traffic to gain information

about traffic shifts for network reconfigurations. To upgrade the cognitive control to the autonomous adaption to current and upcoming network conditions, a traffic estimation based on previous traffic detection results is desired.

3. Monitor the changes of queues when traffic or network reconfiguration changes. This problem will focus on depicting the transient behaviors of the queues in the networks to gain an understanding of how the fast network changes affect the whole systems on a real-time scale.

4. Design the efficient network reconfiguration scheme. With given traffic information and network state information (NSI), the network can reconfigure to improve network performances and save operating costs. Adjustment include both network setting changes (wavelength changes, etc.) and dynamic traffic controls (network load balancing, traffic congestion control, etc.). Continuous assessments on decisions and network controls will be an attribute of the scheme to provide the dynamic optimization of network performance.

5. Evaluate the performance of cognitive design. Questions to be solved include what performance metrics are used and how the cognitive algorithms perform.

6. Estimate the cost of the architecture. Cost evaluations of the network in terms of both capital expenditure (e.g., wavelengths) and operating expenditure (e.g., queueing delay) should be given. Also, recommendations on how to further reduce control complexity should be given.

The architectural recommendation provided in this work will cover all six points above.

1.5 Thesis Scope and Organization

The rest of the thesis is organized as follows.

In Chapter 2, we discuss the traffic model that captures the dynamic traffic environment, the tunneled network architecture, and the reconfiguration options. We build a traffic model based on the characteristics of the dynamic, bursty, and high-granularity traffic. The tunneled network architecture is adopted due to its supremacy

in reducing the control complexity. Three major reconfiguration options in order of preference for the reconfigurable network architecture are introduced.

In Chapter 3, we develop the design of fast-reconfigurable cognitive wavelength management and control algorithms that can accurately adapt based on the traffic conditions in the long coherence time environment, where the traffic changes very slowly. We develop two Bayesian estimators and a stopping-trial estimator to detect traffic changes. We model the transient behaviors of networks to evaluate the detection and queueing delay performances of different estimators. A network cost model is also proposed to stress the trade-off between queueing delay performance and the cost of the capacity plus the control resources used. Finally, the stopping-trial estimator with continuous assessment is recommended due to the fastest response time to traffic changes following by the lowest operating cost.

In Chapter 4, we discuss the fast detection in the moderate coherence time environment, where the traffic changes at a moderate rate. We validate the efficacy of the stopping-trial estimator and discuss its detection performance. As long as the inter-arrival times of traffic sessions are independent but not necessarily identically distributed, the stopping-trial estimator is still recommended as it can react effectively to the traffic rate changes.

In Chapter 5, we discuss traffic trend detection in the short coherence time environment, where the traffic changes very quickly. We model the transient behavior of the network traffic as it drifts towards convergence at a new steady state and validate the feasibility of the traffic trend prediction. Given the fast-changing traffic where the traffic rate changes monotonically in a linear model, we develop the design of predicting the traffic trend with a sequential maximum likelihood estimator based on distribution of the inhomogeneous Poisson process. The algorithm can sufficiently estimate the traffic trend with a reasonable number of arrivals and trigger reconfigurations.

In Chapter 6, we discuss the rerouting algorithm. We develop the rerouting algorithm implemented with the shortest-path algorithm. The rerouting is triggered by a threshold on the queueing delay, and the threshold is directly determined by

the current traffic situation and the network configurations. We also discuss the option to reserve wavelength for rerouted traffic only and do not recommend it due to low resource utilization. The desired properties of paths for rerouting and network topology are discussed. Traffic splitting with resources partitioned in rerouting is recommended for better network utilization.

In Chapter 7, we conclude the thesis with a summary of the recommended architectural design of the cognitive optical networks in the dynamic traffic environments. We also present some directions for future work.

Chapter 2

Dynamic Traffic Environment, Tunneled Architecture, and Reconfigurations

In this chapter, we discuss the simplified traffic model that captures the dynamic traffic environment, the tunneled network architecture, and the reconfiguration options. We build a simplified traffic model to reflect the characteristics of dynamic, bursty, and high-granularity traffic. The tunneled network architecture will be adopted due to its supremacy in reducing the control complexity [55]. In the following chapter, we design the detection and reconfiguration scheme to fulfill the traffic demands based on this traffic model presented in this chapter.

2.1 Dynamic Traffic Model

One of the main contributions of this work is to build a model of the time-varying stochastic traffic in the optical networks, which is usually ignored in the static or quasi-static traffic model in previous work. Other traffic models depicting the time-varying stochastic traffic can be discussed in future work. To capture the bursty nature of traffic, we address the different changes in it. Traffic environments can be categorized into three regimes as follows:

1. Network traffic changes very slowly so adaption can be done while the traffic is in the same state;

2. Network traffic changes at a moderate rate commensurate with the fastest adaption times;

3. Network traffic changes very rapidly so adaptions can only be limited to trends rather than to the detailed changes.

A comprehensive analysis of such dynamic network environments is a must for the design of cognitive management and control of optical networks. In the following chapters, we focus on the design of a traffic detection mechanism and the reconfiguration scheme for all three regimes.

2.1.1 All-to-all Stochastic Traffic

Given a network in the topology form, each node sends a certain amount of traffic to other nodes for transmission. For a well-planned network, the average traffic should be balanced to maintain a low network operating cost [24]. For example, in a Metropolitan Area Network (MAN), the average traffic volume generated in each MAN node should be approximately the same because the network resources are partitioned based on the traffic demands. Several areas with low traffic demands will share the same MAN node, while an area with high traffic demands will be allocated an entire MAN node [24, 58]. Besides, traffic transmission among different node pairs should be considered as statistically independent, since traffic is aggregated from different areas.

In the traffic model, we assume that every source-destination node pair has uniform all-to-all independent and identically distributed (I.I.D.) traffic as shown in Fig. 2-1. The I.I.D. property applies to both the traffic arrival pattern and the traffic transaction size. The I.I.D. arrival pattern comes from the evenly balanced traffic among all the nodes. The I.I.D. traffic transaction size is adopted to facilitate analyses.

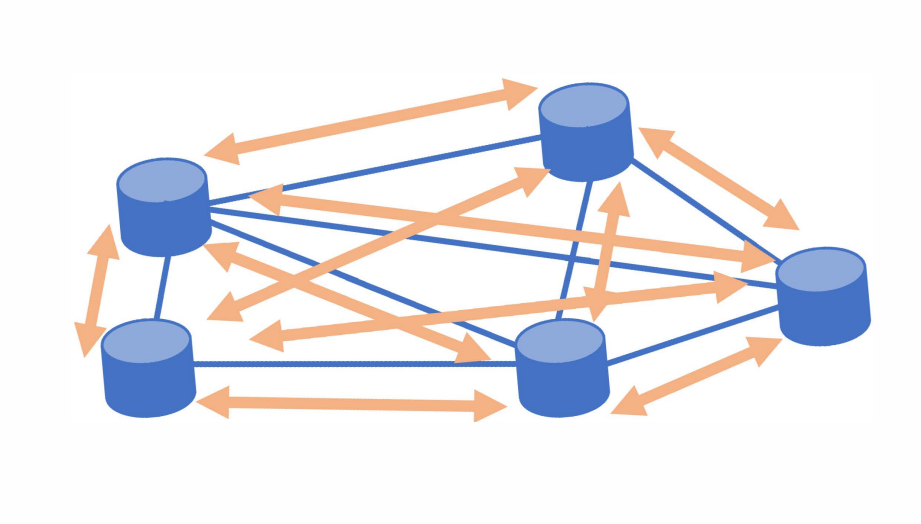

Figure 2-1: All-to-all dynamic traffic.

2.1.2 Multi-state Markov Arrival Rate

The traffic arrivals between each source-destination pair is assumed to follow a doubly stochastic point process with a changing arrival rate $\lambda(t)$. Compared to a fixed arrival rate, the current model with a changing rate can not only cover a wider range of arrival patterns, but capture the burstiness of the traffic. Though the detection and estimation of the changing arrival rate are much harder than the unvarying arrival rate, our model with the changing arrival rate will provide a more generic solution.

We can describe the Poisson arrival process of the traffic in two ways. First, we describe the number of arrivals N in the observation interval. N follows a Poisson distribution with the rate of $\lambda(t)T$. By counting the number of arrivals in the observation time, we can estimate the arrival rate. Denote the observation interval as $[t-T, t]$ with the length of T. To avoid ambiguity, we include both the epochs $(t-T)$ and t. The number of arrivals in the interval $[t-T, t]$ is denoted as $N(T)$ and we have

$$Pr[N(T) = n] = \frac{(\lambda(t)T)^n e^{-\lambda(t)T}}{n!}. \tag{2.1}$$

The expectation of the number of arrivals in the observation time T is

$$E[N(T)] = \lambda(t)T. \tag{2.2}$$

The variance of the number of arrivals in the observation time T is

$$Var[N(T)] = \lambda(t)T. \tag{2.3}$$

Second, we can describe the length of the inter-arrival time and further the total length time of N arrivals. Each inter-arrival time in $\{T_i, i \geq 1\}$ follows an exponential distribution with parameter $\lambda(t)$. We assume the first arrival always happens at the starting time $(t - T)$, and T_i is the inter-arrival time between the i^{th} arrival and the $(i + 1)^{th}$ arrival. The sum of N inter-arrival times $T(N) = \sum_{i=1}^{N} T_i$ follows an Erlang distribution with $\lambda(t)$ as

$$Pr[T(N) = \tau] = \frac{\lambda(t)^N \tau^{N-1} e^{-\lambda(t)\tau}}{(N-1)!}. \tag{2.4}$$

The expectation of the observation length with N number of arrivals in the observation time T is

$$E[T(N)] = \frac{N}{\lambda(t)}. \tag{2.5}$$

The variance of the observation length with N number of arrivals in the observation time T is

$$Var[T(N)] = \frac{N}{\lambda(t)^2} \tag{2.6}$$

We assume $\lambda(t)$ follows a Markov process, where the traffic arrival rate $\lambda(t)$ switches among different states $\lambda_1, \lambda_2, ..., \lambda_l$ as the embedded Markov chain of a countable-state Markov process shown in Fig. 2-2, where $0 < \lambda_1 < \lambda_2 < ... < \lambda_l$. In this model, the traffic arrival rate will not change dramatically in a short amount of time. We assume the transitions only include a state that switches to its previous state, its next state, or itself. The transition from λ_i to $\lambda_{i+1}(0 < i < l-1)$ indicates a traffic surge, and we want to detect it promptly to avoid potential traffic congestion and large queueing delays. The transition from λ_{i+1} to $\lambda_i, (0 < i < l-1)$ indicates a traffic drop, and we also want to detect it promptly to avoid any waste of resources. When $\lambda(t)$ is maintained at state λ_i, there is no traffic change, and we assume the holding interval associated with the state λ_i follows an exponential distribution with a rate of v_i, which is called as the holding rate. Each state is associated with a transition rate $a_{i,j}$ from λ_i to λ_j.

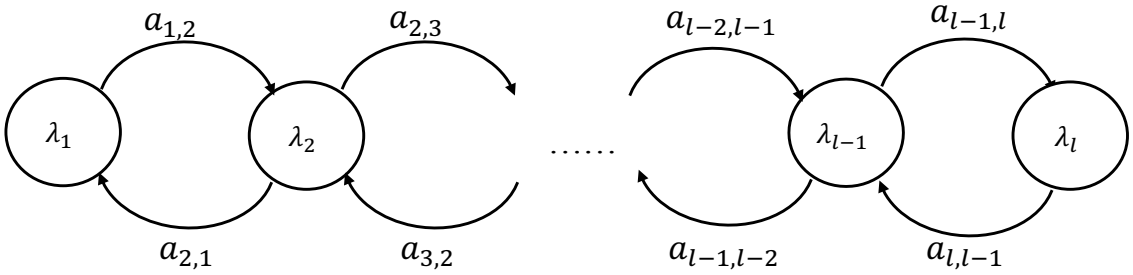

Figure 2-2: Multi-state embedded Markov chain transition of the traffic arrival rate $\lambda(t)$.

Later in Chapter 3, we will simplify the model and assume the traffic switch between a non-surging state λ_0 and a surging state λ_1 to focus on the detection analysis of the traffic changes.

2.1.3 Network Coherence Time

In this work, we model the network coherence time as the holding interval where the traffic arrival rate is maintained at a certain state, and we denote the average coherence time as D_i. The average coherence time of the whole system is denoted as D. In our traffic arrival model, given the current state is λ_i, the average coherence time in state λ_i is

$$D_i = \frac{1}{v_i} = \frac{1}{a_{i,i-1} + a_{i,i+1}}. \tag{2.7}$$

It is because the network will remain in this state, and the time to the next transition is the time until either a transition to the previous state or the next state. With the steady state probability of state λ_i as π_i, the average coherence time for the whole system is

$$D = \sum_{i=1}^{l} \frac{\pi_i}{v_i}. \tag{2.8}$$

We can categorize the network traffic environments using the coherence time D and the time to make decisions into three regimes. Denote the average detection time of the estimator as τ_1. The three regimes of the changing traffic represented in the relation between the detection and the coherence time are:

1. long coherence time: $\tau_1 << D$, where the traffic changes very slowly;
2. moderate coherence time: $\tau_1 \sim D$, where the traffic changes at a moderate rate;
3. short coherence time: $\tau_1 >> D$, where the traffic changes very quickly.

Alternatively, we can quantize the coherence time with the holding rate v_i. We assume v_i is changing within a range of $[v_{min}, v_{max}]$. The range of v_i can be learned and determined from the historical records of network traffic. The three regimes become:

1. long coherence time, where $v_i \sim v_{min}$;

2. moderate coherence time, where $v_i \sim \frac{v_{min}+v_{max}}{2}$;

3. short coherence time, where $v_i \sim v_{max}$.

In the long coherence time regime, the inter-arrival times T_is after the traffic change are a set of I.I.D. variables and the I.I.D. property can facilitate the detection and the estimation of the traffic change. In the moderate coherence time regime, T_is are no longer identically distributed and this makes detection difficult. What is more, the traffic can change again. In the short coherence time regime, it is hard to detect every single traffic change. If the coherence time is extremely short, the detection may no longer converge. In this case, we can only try to predict the trend of fast traffic changes when the session arrival statistics provide enough confidence to do so.

2.2 Tunneled Architecture

We assume a tunneled network architecture, where each node pair is connected via a preselected set of wavelengths within a single lightpath or multiple lightpaths for traffic transmission as shown in Fig. 2-3. Zhang showed in [55] that tunneled architecture can reduce control plane traffic and processing complexity significantly with little sacrifice in efficiency for heavy traffic volumes, compared to meshed architecture that enables full switchability at all nodes. The capacity between each node pair is reconfigurable by adjusting the number of wavelengths used by the node pair based on the offered traffic.

Assume $m(t)$ wavelengths are assigned between a node pair at time t, and each wavelength has a constant transmission rate of R bits per second. The size of each transaction L is assumed to be exponentially distributed with the expectation L_0 in the analysis, but any well-behaved real-life traffic distribution will yield the same architecture recommendations. Given the average size of a transaction L_0, the service rate of each wavelength per transaction is $\mu = \frac{R}{L_0}$.

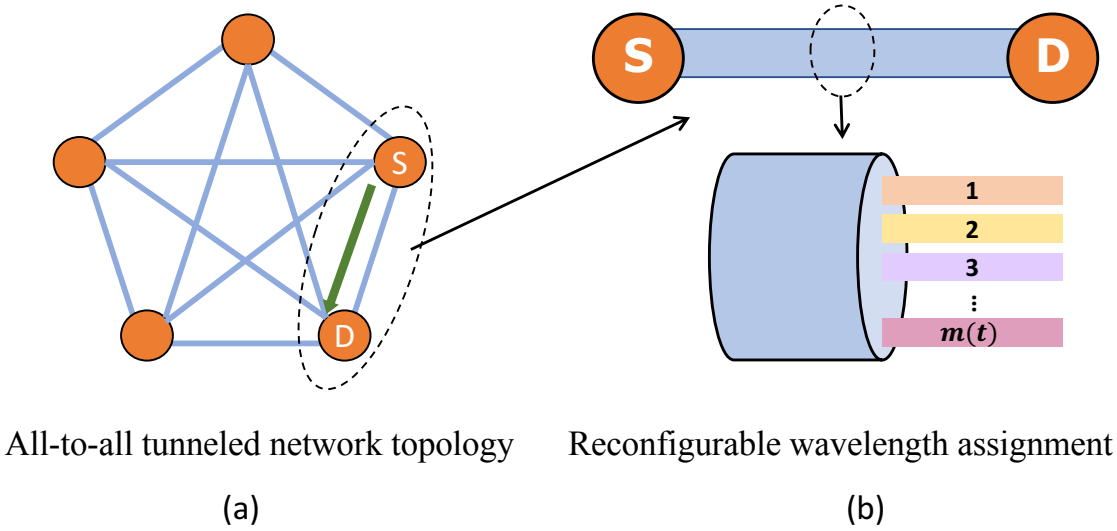

Figure 2-3: An example of an all-to-all tunneled network connection between node pairs in the form of wavelengths. (a) All-to-all tunneled network topology; (b) Reconfigurable wavelength assignment.

2.3 Traffic Load

We employ the traffic load to evaluate the traffic situation in the network. For a node pair, given the arrival rate $\lambda(t)$ and the traffic service μ with $m(t)$ wavelength tunneled in use, we can define the network load between a node pair as

$$\rho = \frac{\lambda(t)}{m(t)\mu}. \tag{2.9}$$

The traffic load of the whole network can be reflected by the load on each node pair. The load of the whole network is the total arrival rate entering the network divided by the total transmission rate of all the wavelengths in use in the network. Given the uniform all-to-all traffic pattern, the average arrival rate of each node pair is the same, and the same number of wavelengths will be assigned to each node pair. Therefore, the average load of the whole network is the same as the average load on

each node pair.

The arrival and transmission between the node pair is in a stable state when $\rho < 1$. When $\rho \geq 1$, the network is overloaded, and the queue between the node pair will grow. Network reconfigurations are needed to bring the system to a new steady state. Otherwise, either long delays or excessive blocking will degrade users' quality of service.

2.4 Reconfigurations

When network traffic changes happen, network reconfigurations may be needed to maintain users' quality of service. An agile network reconfiguration scheme is required to accurately adapt based on the current operating conditions and maintain high cost efficiency. Any mismatch between network conditions and the suggested reconfigurations will increase network burdens and waste network resources.

It is not hard to see that the detection performance of the estimator will determine the following reconfiguration performance since fast detection can avoid the following severe network congestion. Also, high detection accuracy is required since high error probability makes the system unstable and degrades users' quality of service. On the other hand, the network operating cost has to be taken into consideration to provide a reasonable reconfiguration scheme design.

Our goal is to provide timely network reconfigurations in response to the dynamic traffic environments. As discussed in Chapter 1, in the optical networks, with the traffic changes detected, three reconfiguration options in order of preference for the reconfigurable network architecture as shown in Fig. 2-4 are:

1. Lighting up a wavelength in the same primary lightpath between a node pair;

2. Rerouting incremental traffic to a secondary path with open wavelengths between the node pair;

3. Lighting up a new fiber with multiple wavelengths together with optical switching to accommodate overflowing traffic.

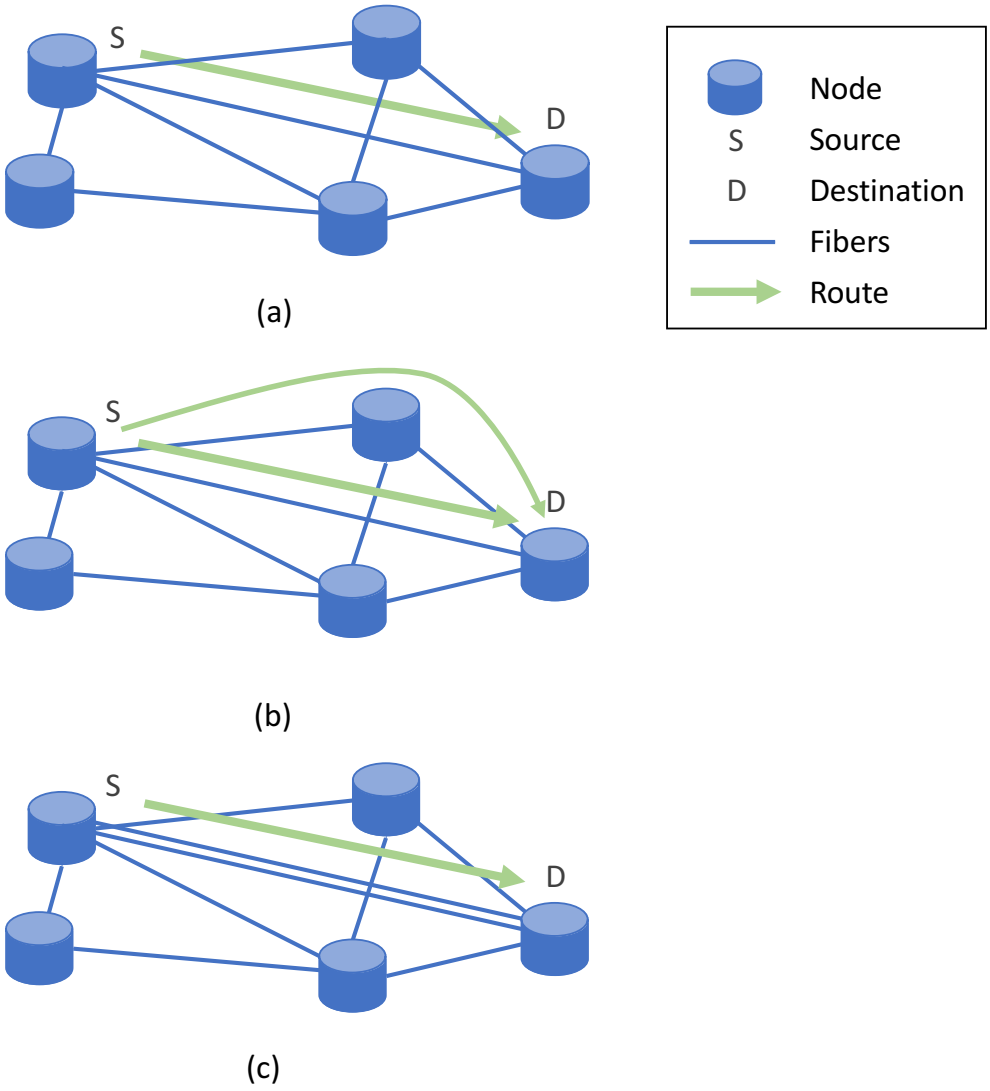

Figure 2-4: The traffic transmission of three network reconfiguration options. (a) The dynamic wavelength addition or subtraction on the primary path. Traffic will be transmitted on the primary path. (b) The rerouting of the incremental traffic. Part of the traffic will be transmitted on paths other than the primary path. (c) The new fiber setup on the primary path. Traffic will be transmitted on the primary path.

The dynamic wavelength addition or subtraction is performed when the maximum number of wavelengths on the path has not been reached. The traffic is still transmitted on the primary path between the source node and the destination node

as shown in Fig. 2-4(a). The fast-reconfigurable cognitive wavelength management and control algorithms aim to accurately adjust the number of assigned wavelengths based on the observation of the operating conditions of the networks. When traffic increases sharply, more wavelengths are expected to be assigned to avoid potential traffic congestion and large queueing delays. When traffic drops, fewer wavelengths are expected to avoid any waste of resources.

When the network traffic reaches a load threshold to satisfy delay requirements, the incremental traffic has to go through other paths to the destination, as shown in Fig. 2-4(b), which is called rerouting. In this work, it is performed when the queueing delay of the traffic transaction exceeds the delay requirement. If the overall system is heavily loaded, a new fiber with multiple wavelengths is needed to be set up, as shown in Fig. 2-4(c), which is equivalent to logically adding a path on the topology. This reconfiguration needs a longer cycle compared to the dynamic wavelength addition/subtraction. We will brief mention the time to perform fiber setup in Chapter 6.

Chapter 3

Long Coherence Time Traffic Environment

In the long coherence time traffic environment, network traffic changes very slowly so adaption can be done while the traffic is in the same state. In other words, we have long enough time for detection and reconfigurations without worrying about more the traffic changes. For the detection of non-stationary traffic changes, we need to find the detection algorithm that can detect the network changes within the shortest possible time when the session arrival statistics provide enough confidence for reconfiguration. An optimal sequential estimator that can make a decision as soon as possible without any predetermined detection parameters is desired [12]. Also, the estimator should rely on less or no historical data and experience to be able to handle rare events effectively and efficiently [12]. This will remedy one of the blind spots of current learning techniques, where historical data is heavily relied on and unanticipated black-swan events are rarely recognized quickly and mitigated on time. The detection algorithm should quickly respond to unanticipated network situations and then avoid successive serious network catastrophes.

In this chapter, we present the design of fast-reconfigurable cognitive wavelength management and control algorithms that can accurately adapt by observing the network operating conditions of the networks. Two Bayesian estimators and a stopping-trial estimator were proposed in [10, 12] to detect traffic changes. The Bayesian

estimator with the fixed observation time to classify the traffic state of the node is analyzed in [31]. Its average packet delay at a given node for various observation times is also given in [31]. In this work, we further develop these estimators and examine their traffic detection and queueing delay performances based on the resulting transient behaviors of networks. A network cost model is proposed to capture the trade-off between reconfiguration performance (transient queueing delays) and the cost of the capacity plus the control resources used. We recommend a wavelength reconfiguration algorithm based on the stopping-trial estimator with continuous assessment where the system reconfigures whenever necessary. The reconfiguration can involve addition or subtraction of multiple wavelengths. Among the three estimators, the stopping-trial estimator requires the smallest number of wavelengths to be reconfigured because it responds the fastest and avoids a high peak queueing delay [59].

The arrival traffic at the source is assumed to form a doubly stochastic Poisson point process with a time-dependent rate of $\lambda(t)$ as mentioned in Chapter 2. In [31], the traffic arrival model is assumed to transit between a surging state and a non-surging state. To facilitate the analysis in this chapter, we use a similar idea to assume $\lambda(t)$ switches between a non-surging state λ_0 and a surging state λ_1, where $\lambda_0 < \lambda_1$ as shown in Fig.3-1. When $\lambda(t)$ switches from λ_0 to λ_1, there is a traffic surge, and we want to detect it promptly to avoid potential traffic congestion and large queueing delays. When $\lambda(t)$ switches from λ_1 to λ_0, there is a traffic drop, and we want to detect it promptly to avoid any waste of resources. The size of each transaction L is exponentially distributed with the expectation L_0 as defined in Chapter 2.

3.1 Two Bayesian Estimators

We first consider the commonly-used Bayesian estimators to detect the changes of traffic statistics. Given the binary nature of $\lambda(t)$, two possible hypotheses for the decision are [31]:

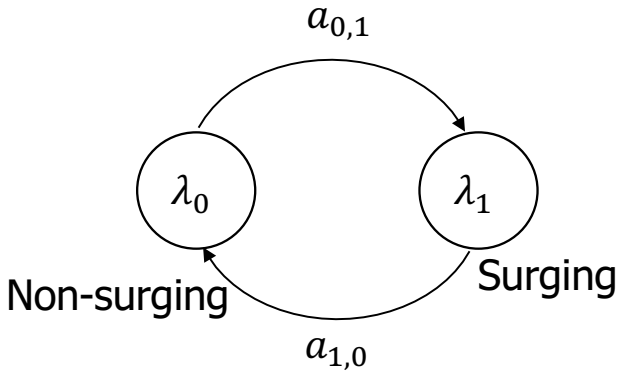

Figure 3-1: Two-state embedded Markov chain transition of the traffic arrival rate $\lambda(t)$.

$$H_0 : \lambda(t) = \lambda_0; \tag{3.1}$$

$$H_1 : \lambda(t) = \lambda_1. \tag{3.2}$$

The false alarm probability P_f is the probability that we accept H_1 when H_0 is true. The missed detection probability P_m is the probability that we accept H_0 when H_1 is true. The probability of detection is $P_d = 1 - P_m$. If the *a priori* probabilities are known as π_{λ_0} for H_0 and $\pi_{\lambda_1} = 1 - \pi_{\lambda_0}$ for H_1, the total error probability is

$$\begin{aligned} Pr[e] &= \pi_{\lambda_0} Pr[H_1|H_0] + \pi_{\lambda_1} && (3.3) \\ &= Pr[H_0|H_1] = \pi_{\lambda_0} P_f + \pi_{\lambda_1} P_m. && (3.4) \end{aligned}$$

We can observe the Poisson arrival process in two ways. First, we observe the number of arrivals N in the observation interval $[t - T, t]$. N follows a Poisson distribution with the rate of $\lambda(t)T$. Notice that T should be less than the network coherence time for effective adaptions in the long coherence time traffic environment. Second, each inter-arrival time in $\{T_i, i \geq 1\}$ follows an exponential distribution with

parameter $\lambda(t)$. As assumed in Chapter 2, the first arrival always happens at the starting time $(t-T)$, and T_i is the interarrival time between the i^{th} arrival and the $(i+1)^{th}$ arrival. The sum of N inter-arrival times $\sum_{i=1}^{N} T_i$ follows an Erlang distribution with $\lambda(t)$.

Given the two ways of observing the arrival process, we can develop two Bayesian estimators fixed-time estimator $\hat{\lambda}_T(t)$ and fixed-count estimator $\hat{\lambda}_N(t)$ as follows.

3.1.1 Fixed-Time Estimator $\hat{\lambda}_T(t)$

For the fixed-time estimator $\hat{\lambda}_T(t)$, we count the total number of arrivals in a fixed time interval $[t-T, t]$ denoted by $N(T)$ backwards in time to determine the validation of the hypotheses [12]. The observation window is moving with time. Here, N is a random variable with the constant parameter T. We define [12]

$$\hat{\lambda}_T(t) = \frac{N(T)}{T}. \tag{3.5}$$

A similar analysis of the Bayesian estimator with the fixed observation time to classify the traffic state of the node with the arrivals in Poisson distribution as shown in [31].

Bayesian Likelihood Ratio Test

Similar to the Bayesian likelihood ratio test in [31], the Bayesian likelihood ratio test for $N(T)$ with *a priori* probabilities π_{λ_0} and π_{λ_1} is

$$\frac{Pr[N(T)=n|\lambda_1]}{Pr[N(T)=n|\lambda_0]} \underset{\lambda_0}{\overset{\lambda_1}{\gtrless}} \frac{\pi_{\lambda_0}}{\pi_{\lambda_1}}, \tag{3.6}$$

where $Pr[N(T)=n|\lambda_0] = \frac{(\lambda_0 T)^n e^{-\lambda_0 T}}{n!}$, $Pr[N(T)=n|\lambda_1] = \frac{(\lambda_1 T)^n e^{-\lambda_1 T}}{n!}$.

We get

$$\Leftrightarrow n \underset{\lambda_0}{\overset{\lambda_1}{\gtrless}} \frac{(\lambda_1 - \lambda_0)T + \ln\left(\frac{\pi_{\lambda_0}}{\pi_{\lambda_1}}\right)}{\ln\left(\frac{\lambda_1}{\lambda_0}\right)} \triangleq \gamma_T. \tag{3.7}$$

Therefore, we observe the number of arrivals $N(T)$ in $[t-T, t]$. If $N(T) \geq \gamma_T$, we accept H_1 and reject H_0. If $N(T) < \gamma_T$, we accept H_0 and reject H_1.

Similar to the Bayesian likelihood ratio test in [31], when all the *a priori* probabilities are the same, *i.e.,* $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5$, we have the test as

$$n \underset{\lambda_0}{\overset{\lambda_1}{\gtrless}} \frac{(\lambda_1 - \lambda_0)T}{\ln\left(\frac{\lambda_1}{\lambda_0}\right)} \triangleq \gamma_T. \tag{3.8}$$

The error probability $Pr[e_T]$ for $\hat{\lambda}_T(t)$ is

$$\begin{aligned}
Pr[e_T] &= \pi_{\lambda_0} Pr[H_1|H_0] + \pi_{\lambda_1} Pr[H_0|H_1] && (3.9)\\
&= \pi_{\lambda_0} Pr[N(T) \geq \gamma_T | \lambda(t) = \lambda_0] + \pi_{\lambda_1} Pr[N(T) < \gamma_T | \lambda(t) = \lambda_1] && (3.10)\\
&= \pi_{\lambda_0} \sum_{n=\lceil \gamma_T \rceil}^{\infty} Pr[N(T) = n|\lambda_0] + \pi_{\lambda_1} \sum_{n=0}^{\lceil \gamma_T \rceil - 1} Pr[N(T) = n|\lambda_1] && (3.11)\\
&= \pi_{\lambda_0} e^{-\lambda_0 T} \sum_{n=\lceil \gamma_T \rceil}^{\infty} \frac{(\lambda_0 T)^n}{n!} + \pi_{\lambda_1} e^{-\lambda_1 T} \sum_{n=0}^{\lceil \gamma_T \rceil - 1} \frac{(\lambda_1 T)^n}{n!}. && (3.12)
\end{aligned}$$

Neyman-Pearson Test

The Bayesian likelihood ratio test requires the knowledge of all the *a priori* probabilities. In practice, a more general case is that all the *a priori* probabilities are unknown. For the Neyman-Pearson test, it assumes all the *a priori* probabilities are unknown. As mentioned in [31], instead, define a threshold η for the likelihood ratio test and we have

$$\frac{Pr[N(T)=n|\lambda_1]}{Pr[N(T)=n|\lambda_0]} \underset{\lambda_0}{\overset{\lambda_1}{\gtrless}} \eta, \tag{3.13}$$

As mentioned in [31], similar to the expression in 3.7, the threshold expression is

$$n \underset{\lambda_0}{\overset{\lambda_1}{\gtrless}} \frac{(\lambda_1-\lambda_0)T+\ln\eta}{\ln\left(\frac{\lambda_1}{\lambda_0}\right)} \triangleq \gamma_{T'}. \tag{3.14}$$

A Gaussian approximation to compute the error probabilities (and to create an ROC curve) instead of the exact error probabilities is shown in [31]. In this work, we prove the exact error probabilities as follows.

The type 1 error probability (false alarm probability) P_{f_T} is the probability that we accept H_1 but actually H_0 is true, which is the same idea as $Pr[H_1|H_0]$ in the Bayesian likelihood ratio test. Equivalently, it is the probability that we report $\lambda(t)=\lambda_1$ given $\lambda(t)$ is actually in the level of λ_0. Since n is an integer, we have

$$P_{f_T} = Pr[H_1|H_0] = \sum_{n=\lceil\gamma_{T_N}\rceil}^{\infty} Pr[N(T)=n|\lambda_0] \tag{3.15}$$

$$= \sum_{n=\lceil\gamma_{T'}\rceil}^{\infty} \frac{(\lambda_0 T)^n e^{-\lambda_0 T}}{n!}, \tag{3.16}$$

where $\lceil\cdot\rceil$ is the ceiling function.

The type 2 error probability (missed detection probability) P_{m_T} is the probability that we accept H_0 but actually H_1 is true, which is the same idea as $Pr[H_0|H_1]$ in the Bayesian likelihood ratio test. Equivalently, it is the probability that we report $\lambda(t)=\lambda_0$ given $\lambda(t)$ is actually in the level of λ_1. Since n is an integer, we have

$$P_{m_T} = Pr[H_0|H_1] = \sum_{n=0}^{\lceil \gamma_{T_N} \rceil - 1} Pr[N(T) = n|\lambda_1] \tag{3.17}$$

$$= \sum_{n=0}^{\lceil \gamma_{T'} \rceil - 1} \frac{(\lambda_1 T)^n e^{-\lambda_1 T}}{n!}. \tag{3.18}$$

We can use Chernoff bounds to approximate the false alarm probability and the missed detection probability. However, the detection time T after a rate change needs to be as short as possible to achieve the fast response that prevents the queue build-up. A short detection time will inevitably cause higher false alarm/missed detection rates, where the exponentially tight Chernoff bound is not a good approximation. Though we will not use the Chernoff bound approximation in this work, it does provide an easily calculable approximation when the requirement of the probability of false alarm/missed detection is strict.

3.1.2 Fixed-Count Estimator $\hat{\lambda}_N(t)$

The fixed-time estimator requires a fixed observation time, which may not be able to adapt well to fast-changing traffic. A fixed-count estimator has been proposed in [10, 12] to enable the flexible observation time with the fixed number of counts. Here, we further elaborate on this fixed-count estimator $\hat{\lambda}_N(t)$. For the fixed-count estimator $\hat{\lambda}_N(t)$, we observe the duration $T(N)$, formed by the last N arrivals (including the one at $(t-T)$) backwards in time to determine the validation of the hypotheses [12]. We define the fixed-count estimator as [12]

$$\hat{\lambda}_N(t) = \frac{N}{T(N)} = \frac{N}{\sum_{i=1}^{N-1} T_i + Z(t)}, \tag{3.19}$$

where $Z(t)$ is the age of the Poisson process of the observation interval ending at time t, which is defined as the interval from the most recent arrival (N^{th} arrival) before (but not including) t until t. Namely, $Z(t)$ is the interval from the N^{th} arrival until

the epoch t. If the N^{th} arrival happens at time t, $Z(t) = 0$. $(N-1)$ inter-arrivals are included in the previous N arrivals, so that

$$Z(t) = T - \sum_{i=1}^{N-1} T_i. \tag{3.20}$$

We can prove $Z(t)$ also follows an exponential distribution with rate $\lambda(t)$. If we look at the arrivals of the Poisson process in $[t-T, t]$ backward in time, it is still a Poisson process due to its time-reversibility [21]. For the reversed Poisson process, $Z(t)$ becomes the interval between a starting epoch (the epoch t when looking forward in time)and the first arrival (the N^{th} arrival when looking forward in time). Due to the memoryless property of exponential distribution, $Z(t)$ follows an exponential distribution with the arrival rate of $\lambda(t)$ as the following inter-arrival intervals. Therefore, we have

$$\hat{\lambda}_N(t) = \frac{N}{T(N)} = \frac{N}{\sum_{i=1}^{N} T_i}. \tag{3.21}$$

Bayesian Likelihood Ratio Test

Similarly, the Bayesian likelihood ratio test for $T(N)$ with *a priori* probabilities $Pr[H_0] = \pi_{\lambda_0}$ and $Pr[H_1] = \pi_{\lambda_1}$ is

$$\frac{f_{T|\lambda}(\tau|\lambda_1)}{f_{T|\lambda}(\tau|\lambda_0)} \underset{\lambda_0}{\overset{\lambda_1}{\gtrless}} \frac{\pi_{\lambda_0}}{\pi_{\lambda_1}}, \tag{3.22}$$

where $f_{T|\lambda}(\tau|\lambda_1) = \frac{\lambda_1^N \tau^{N-1} e^{-\lambda_1 \tau}}{(N-1)!}$, $f_{T|\lambda}(\tau|\lambda_0) = \frac{\lambda_0^N \tau^{N-1} e^{-\lambda_0 \tau}}{(N-1)!}$.

We get

$$\tau \underset{\lambda_1}{\overset{\lambda_0}{\lessgtr}} \frac{\ln\left(\frac{\pi_{\lambda_1}}{\pi_{\lambda_0}}\right) + N \ln\left(\frac{\lambda_1}{\lambda_0}\right)}{(\lambda_1 - \lambda_0)} \triangleq \gamma_N. \tag{3.23}$$

Therefore, we observe the length of time interval $T(N)$ for N arrivals. If $T(N) \geq \gamma_N$, we accept H_0 and reject H_1; if $T(N) < \gamma_N$, we accept H_1 and reject H_0.

When all the *a priori* probabilities are the same, *i.e.,* $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5$, we have the threshold as

$$\tau \underset{\lambda_1}{\overset{\lambda_0}{\gtrless}} \frac{N \ln\left(\frac{\lambda_1}{\lambda_0}\right)}{(\lambda_1 - \lambda_0)} \triangleq \gamma_N. \tag{3.24}$$

Similarly, the error probability of decision making $Pr[e_N]$ for $\hat{\lambda}_N(t)$ is

$$\begin{aligned}
Pr[e_N] &= \pi_{\lambda_0} Pr[H_1|H_0] + \pi_{\lambda_1} Pr[H_0|H_1] && (3.25)\\
&= \pi_{\lambda_0} Pr[T(N) < \gamma_N | \lambda(t) = \lambda_0] + \pi_{\lambda_1} Pr[T(N) \geq \gamma_N | \lambda(t) = \lambda_1] && (3.26)\\
&= \pi_{\lambda_0} \int_0^{\gamma_N} f_{T|\lambda}(\tau|\lambda_0) d\tau + \pi_{\lambda_1} \int_{\gamma_N}^{\infty} f_{T|\lambda}(\tau|\lambda_1) d\tau && (3.27)\\
&= \pi_{\lambda_0} \int_0^{\gamma_N} \frac{\lambda_0^N \tau^{N-1} e^{-\lambda_0 \tau}}{(N-1)!} d\tau + \pi_{\lambda_1} \int_{\gamma_N}^{\infty} \frac{\lambda_1^N \tau^{N-1} e^{-\lambda_1 \tau}}{(N-1)!} d\tau && (3.28)\\
&= \frac{\pi_{\lambda_0} \lambda_0^N}{(N-1)!} \int_0^{\gamma_N} \tau^{N-1} e^{-\lambda_0 \tau} d\tau + \frac{\pi_{\lambda_1} \lambda_1^N}{(N-1)!} \int_{\gamma_N}^{\infty} \tau^{N-1} e^{-\lambda_1 \tau} d\tau. && (3.29)
\end{aligned}$$

Neyman-Pearson Test

The Neyman-Pearson test assumes all the *a priori* probabilities are unknown. Define a threshold η, and we have

$$\frac{f_{T|\lambda}(\tau|\lambda_1)}{f_{T|\lambda}(\tau|\lambda_0)} \underset{\lambda_0}{\overset{\lambda_1}{\gtrless}} \eta. \tag{3.30}$$

Similar to the expression in 3.23, the threshold expression is

$$\tau \underset{\lambda_1}{\overset{\lambda_0}{\gtrless}} \frac{N \ln\left(\frac{\lambda_1}{\lambda_0}\right) - \ln \eta}{(\lambda_1 - \lambda_0)} \triangleq \gamma_{N'}. \tag{3.31}$$

The type 1 error probability (false alarm probability) P_{f_T} is the probability that we accept H_1 but actually H_0 is true, which is the same idea as $Pr[H_1|H_0]$ in the Bayesian LRT. Equivalently, it is the probability that we report $\lambda(t) = \lambda_1$ given $\lambda(t)$ is actually in the level of λ_0. We have

$$P_{f_N} = Pr[H_1|H_0] = \int_0^{\gamma_{N'}} f_{T|\lambda}(\tau|\lambda_0)d\tau \tag{3.32}$$

$$= \int_0^{\gamma_{N'}} \frac{\lambda_0^N \tau^{N-1} e^{-\lambda_0 \tau}}{(N-1)!} d\tau. \tag{3.33}$$

The type 2 error probability (missed detection probability) P_{m_T} is the probability that we accept H_0 but actually H_1 is true, which is the same idea as $Pr[H_0|H_1]$ in the Bayesian LRT. Equivalently, it is the probability that we report $\lambda(t) = \lambda_0$ given $\lambda(t)$ is actually in the level of λ_1. We have

$$P_{m_N} = Pr[H_0|H_1] = \int_{\gamma_{N'}}^{\infty} f_{T|\lambda}(\tau|\lambda_1)d\tau \tag{3.34}$$

$$= \int_{\gamma_{N'}}^{\infty} \frac{\lambda_1^N \tau^{N-1} e^{-\lambda_1 \tau}}{(N-1)!} d\tau. \tag{3.35}$$

3.1.3 Comparison of the Two Bayesian Estimators

The simulated rate change detection shapes of both $\hat{\lambda}_T(t)$ and $\hat{\lambda}_N(t)$ in a single run is shown in Fig. 3-2. No assumption of $\lambda(t)$ is made to better demonstrate the detection process. The arrival samples are also plotted to better describe the detection process. The fixed count N in $\hat{\lambda}_N(t)$ is chosen such that $N = \lambda_0 T$ for the fixed time T in $\hat{\lambda}_T(t)$. Due to the different detection mechanisms, the shape of the detection curves of $\hat{\lambda}_T(t)$ and $\hat{\lambda}_N(t)$ are different. $\hat{\lambda}_T(t)$ will not change if the number of arrivals is not changing in the moving observation time window, and the shape looks like a series of steps. $\hat{\lambda}_N(t)$ will continuously decay if no arrival comes, and it will go up when a new arrival is detected. From the shape, we can find $\hat{\lambda}_N(t)$ is more sensitive to the change than $\hat{\lambda}_T(t)$. Besides, $\hat{\lambda}_N(t)$ has a non-fixed observation time compared to

$\hat{\lambda}_T(t)$, which has a fixed observation time window.

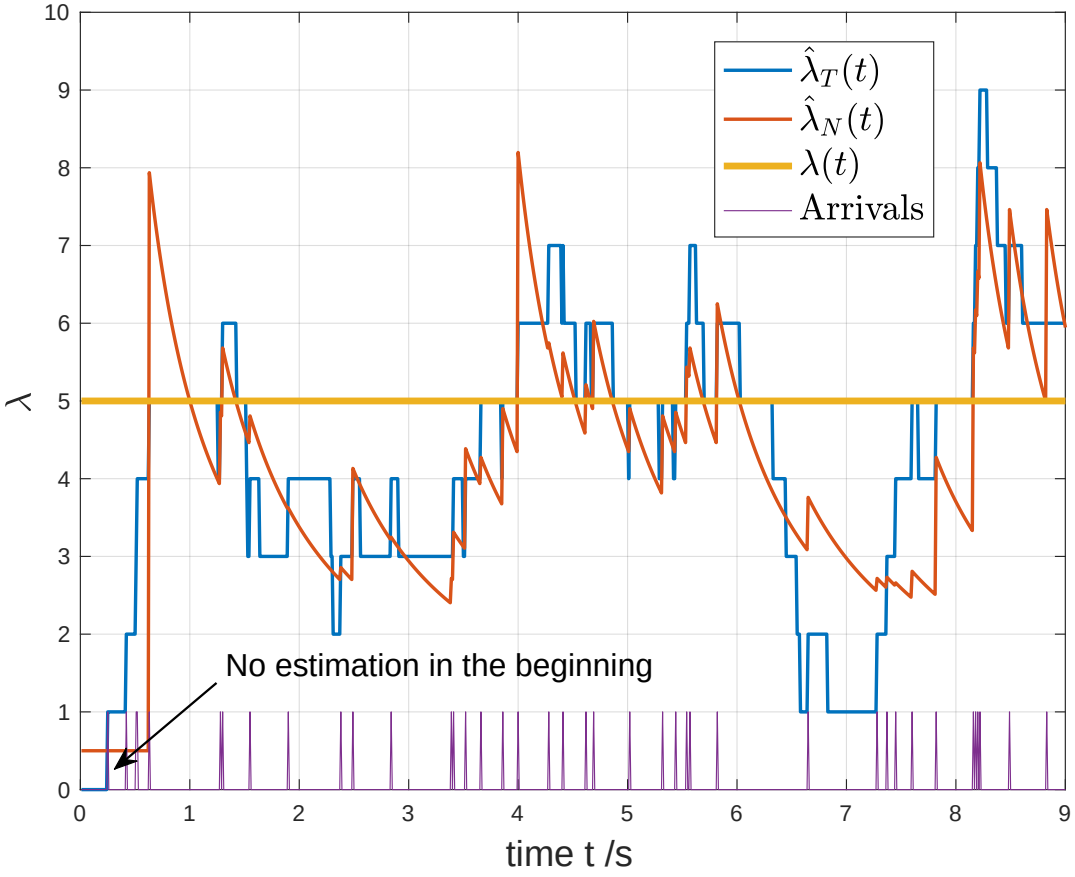

Figure 3-2: The comparison of detection shapes of $\hat{\lambda}_T(t)$ and $\hat{\lambda}_N(t)$ in a single run. No assumption of $\lambda(t)$ is made. $\lambda_0 = 5$ $T = 1, N = \lambda_0 T = 5$.

The simulated rate change detection performance comparisons of both $\hat{\lambda}_T(t)$ and $\hat{\lambda}_N(t)$ with a step change in $\lambda(t)$ in the average of different numbers of runs (1 run, 10 runs, 100 runs, and 1000 runs) are shown in Fig. 3-3. The simulated rate change detection performance comparisons of both $\hat{\lambda}_T(t)$ and $\hat{\lambda}_N(t)$ with a step change in $\lambda(t)$ in the average of 10000 runs are shown in Fig. 3-4. No assumption of $\lambda(t)$ is made to better demonstrate the detection process. The probability distribution is the same, but each time the samples will be randomly generated, which incurs the zig-zag shapes of the detection results. The fixed count N in $\hat{\lambda}_N(t)$ is chosen such that $N = \lambda_0 T$

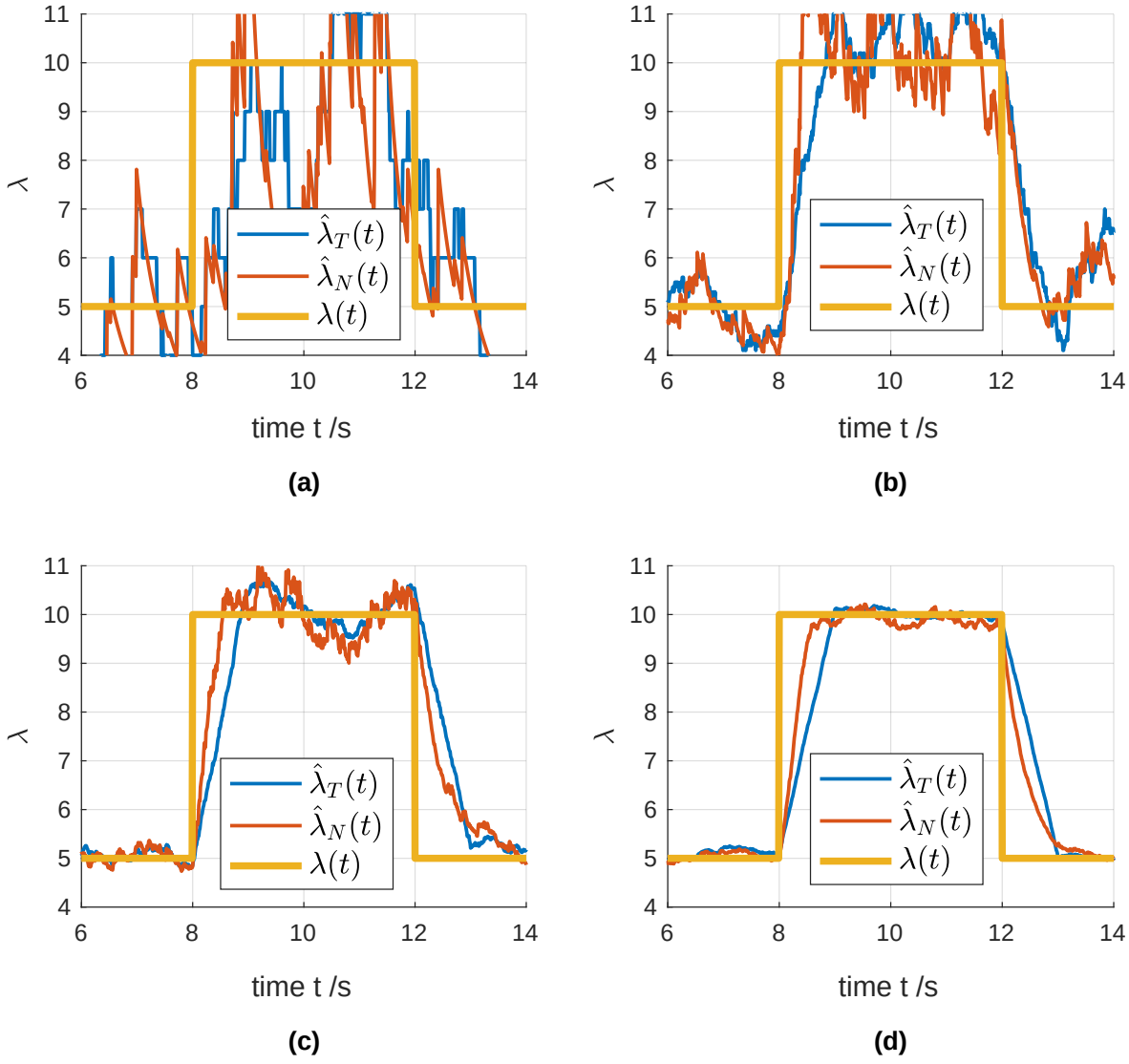

Figure 3-3: The comparison of the average detection results over different numbers of runs of $\hat{\lambda}_T(t)$ and $\hat{\lambda}_N(t)$: (a) a single run; (b) 10 runs; (c) 100 runs; (d) 1000 runs. No assumption of $\lambda(t)$ is made. The probability distribution of $\lambda(t)$ is the same but each time the arrival samples will be randomly generated. $\lambda_0 = 5, \lambda_1 = 10$. $T = 1, N = \lambda_0 T = 5$.

for the fixed time T in $\hat{\lambda}_T(t)$. From it, we find that a single-run result does not well show the catch of the step change. The average of more runs gives a smoother results. With the average results of over 10000 runs as shown in Fig. 3-4, both estimators can successfully estimate the non-varying arrival rates. When a step traffic change of

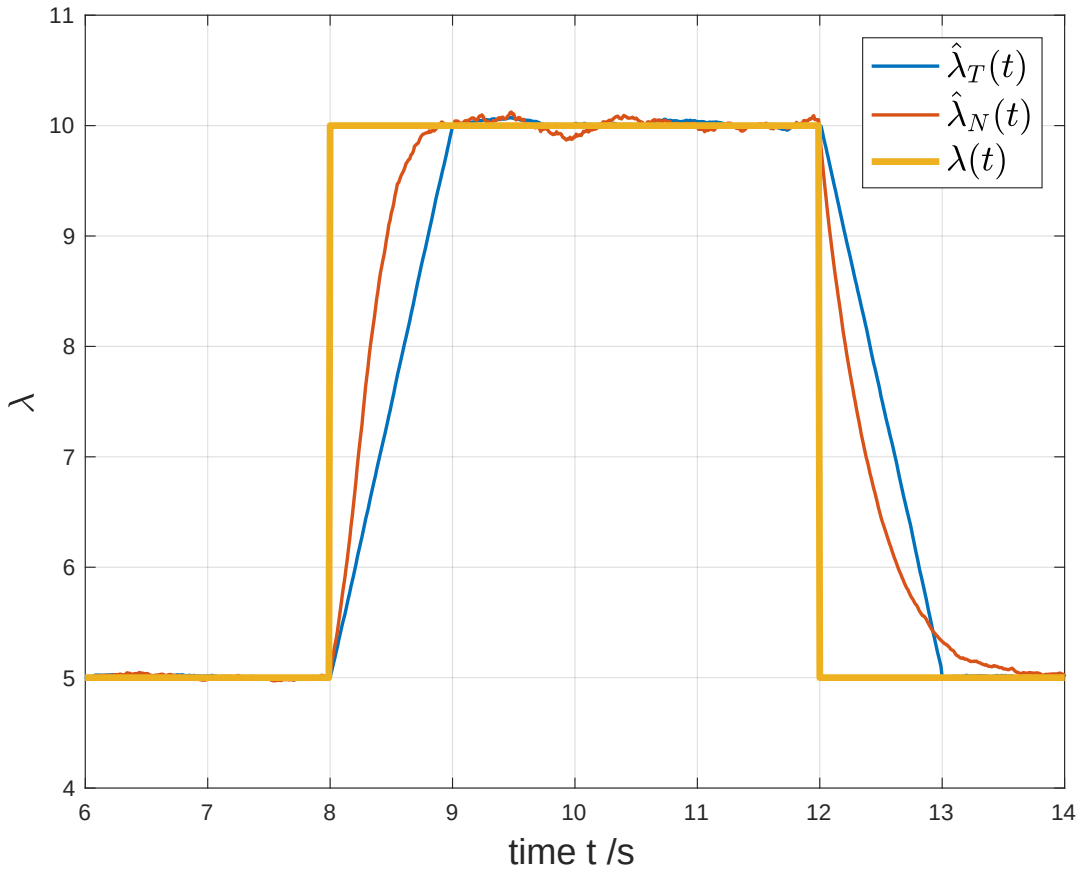

Figure 3-4: The comparison of the average detection results over 10000 runs of $\hat{\lambda}_T(t)$ and $\hat{\lambda}_N(t)$. No assumption of $\lambda(t)$ is made. The probability distribution of $\lambda(t)$ is the same, but each time the arrival samples will be randomly generated. $\lambda_0 = 5, \lambda_1 = 10$. $T = 1, N = \lambda_0 T = 5$.

$\lambda(t)$ (either traffic surge or traffic drop) happens, $\hat{\lambda}_T(t)$ always takes a time of T to move to the new state as it uses the fixed observation time. $\hat{\lambda}_N(t)$ always responds faster to sudden rate changes than $\hat{\lambda}_T(t)$, though $\hat{\lambda}_N(t)$ may not be as accurate and stable as $\hat{\lambda}_T(t)$, which is also shown in the receiver operating characteristic (ROC) comparisons of the two estimators in Fig. 3-5. From the ROC curves, given the same probability of false alarm, we find that $\hat{\lambda}_N(t)$ has a worse probability of detection than that of $\hat{\lambda}_T(t)$ with the fixed count of $\hat{\lambda}_N(t)$ set to $N = \lambda_0 T$, where T is the length of the fixed time observation interval. Compared to the fixed time T, the detection time of $\hat{\lambda}_N(t)$ can flexibly adjust to the underlying $\lambda(t)$. In the setting of the step

change in Fig. 3-3 and Fig. 3-4, $\hat{\lambda}_N(t)$ can improve network efficiency by avoiding both a long detection time for fast changes and a high sampling frequency for low arrival rates. These simulation results indicate that $\hat{\lambda}_N(t)$ is superior for this arrival rate changing scenario, but that may not be true if there is less separation in states than here. Also, if the penalty of the error probability is high, $\hat{\lambda}_N(t)$'s advantage in fast detection time may not be desirable.

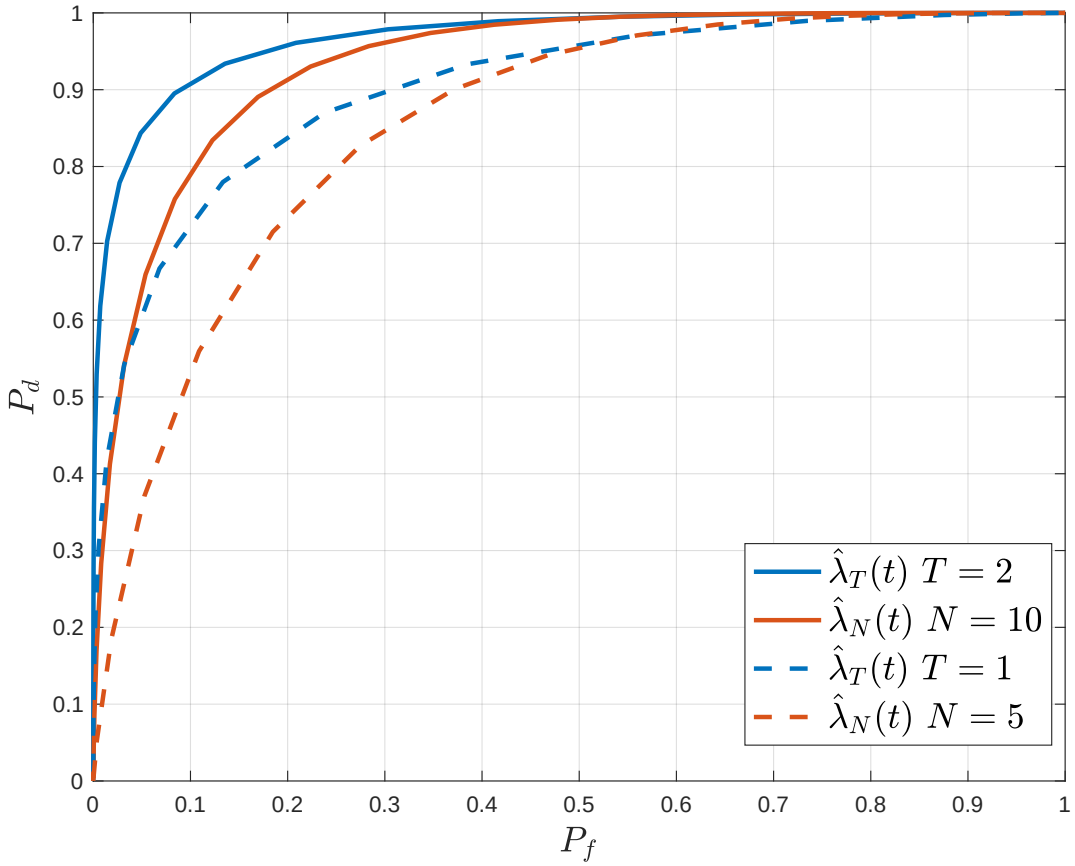

Figure 3-5: The comparison of ROC curves of fixed-time estimator $\hat{\lambda}_T(t)$ and fixed-count estimator $\hat{\lambda}_N(t)$. $\lambda_0 = 5, \lambda_1 = 10$. The fixed count N in $\hat{\lambda}_N(t)$ is chosen such that $N = \lambda_0 T$ for the fixed time T in $\hat{\lambda}_T(t)$.

Though both Bayesian estimators are simple to implement, they have the disadvantage of taking too much time to react when there are more arrivals than expected and reacting to noisy data when there are not enough arrivals to give a good estimate

of the underlying rate change. In both cases, this leads to inappropriate and even disruptive reconfigurations.

Another major drawback is that both Bayesian estimators require the *a priori* probability distribution π_{λ_0} and π_{λ_1}, which are not usually known and also can be non-stationary. This can be estimated over the coherence time of the network from prior traffic statistics using learning techniques. The caution with such learning techniques is that it will not do well against an extremely rare or black-swan event. The assumption, that the coherence time of the arrival process is longer than the times that reconfigurations occur, is the regime where such algorithms are useful. In the event that the coherence time is shorter than the times needed for reconfiguration, adaptive techniques will not be effective. Therefore, we need to find an algorithm that runs continuously, and adapts the system at any time scale and achieves the delay performance of the lower envelope of the various estimators. We will explore the efficacy of a sequential decision algorithm, the stopping-trial estimator, in the next section.

3.2 Stopping-Trial Estimator

An estimator called the stopping-trial estimator $\hat{\lambda}_{ST}(t)$ is proposed in [10, 12] to detect traffic arrival rate changes in the shortest possible time so that the system can be reconfigured at a fast time scale. For the stopping-trial estimator $\hat{\lambda}_{ST}(t)$, we observe each inter-arrival time T_i of the doubly stochastic Poisson point process as a sequential test to trigger network reconfigurations. As opposed to $\hat{\lambda}_T(t)$ and $\hat{\lambda}_N(t)$, $\hat{\lambda}_{ST}(t)$ does not require a predetermined observation time or count. It can make a decision in the shortest possible time when the session arrival statistics provide enough confidence for reconfiguration [21].

We elaborate the design of the stopping-trial estimator $\hat{\lambda}_{ST}(t)$. The detection process of $\hat{\lambda}_{ST}(t)$ can be modeled as a random walk S_J based on $\{T_i, i \geq 1\}$, where J is the time that a threshold is crossed and a reconfiguration is made. Based on results in [12], if the process starts from a non-surging state, the random walk can be

defined as

$$S_J = \sum_{i=1}^{J} (T_i - \frac{1}{\lambda_0}). \tag{3.36}$$

Define the random walk if the process starts from a surging state as

$$S_J = \sum_{i=1}^{J} (T_i - \frac{1}{\lambda_1}). \tag{3.37}$$

To avoid bias from the previous decision, the algorithm resets S_J and starts from the new state once a traffic rate change is detected. Two sample random walks are shown in Fig. 3-6 [12], where n is a discretized time index in the unit of arrivals. Denote the threshold for adding a new wavelength as η_+ and the threshold for tearing down an existing wavelength as η_-. Both η_+ and η_- are determined by the desired error probabilities. The wavelength reconfiguration algorithm for two states is shown in Algorithm 1. A generalized stopping-trial estimator $\hat{\lambda}_{ST}(t)$ for the multi-state Markov chain will be discussed in Chapter 4.

We can get an exponentially tight upper bound of the missed detection probability for a surge from Wald's identity in [21] is

$$P_{m_{ST}} \quad \leq e^{-r^* \eta_-}, \tag{3.38}$$

where r^* is the positive root for the semi-invariant moment generation function (MGF) of the step variable $T_i - \frac{1}{\lambda_0}$ as

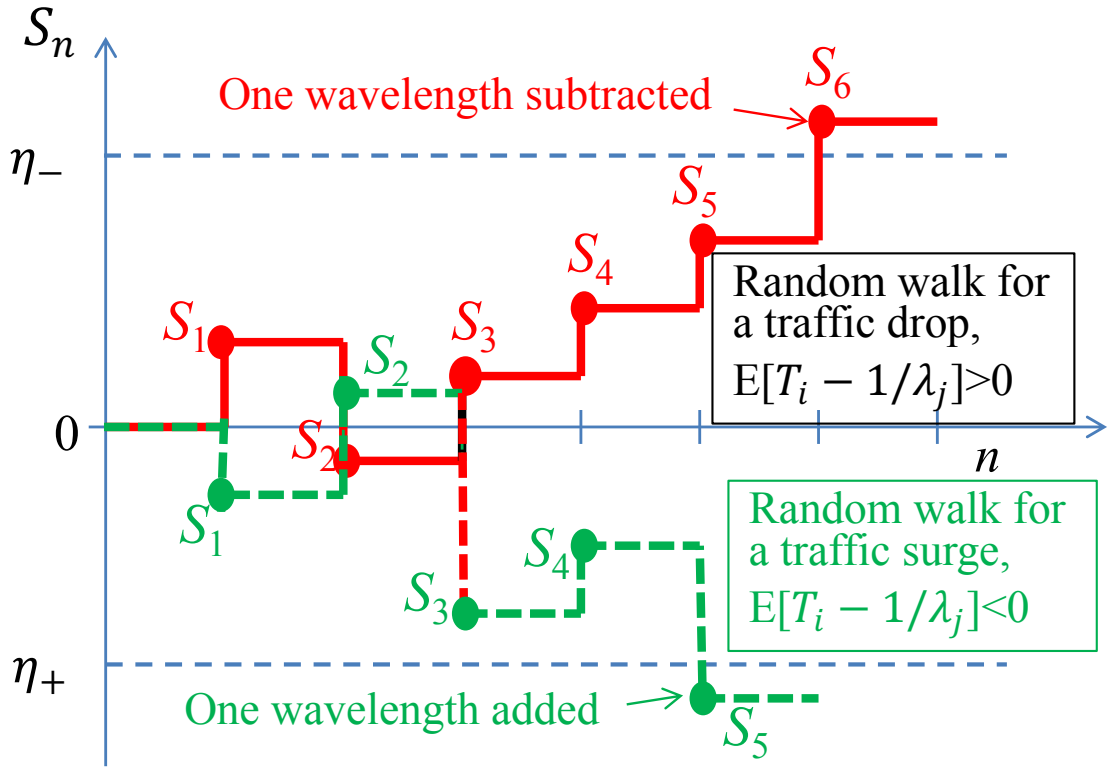

Figure 3-6: Sample functions of random walks S_n with one for traffic surge and one for traffic drop [12]. n is a discretized time index in the unit of arrivals.

$$\ln(E[e^{r(T_i - \frac{1}{\lambda_0})}]) = 0 \tag{3.39}$$

$$\Leftrightarrow \ln\left[\int_0^\infty e^{r(t-\frac{1}{\lambda_0})} \cdot \lambda_1 e^{-\lambda_1 t} dt\right] = 0 \tag{3.40}$$

$$\Leftrightarrow \lambda_1 e^{-\frac{r}{\lambda_0}} + r - \lambda_0 = 0, \tag{3.41}$$

where r is valid when $r < \lambda_1$.

Algorithm 1 Stopping-trial wavelength reconfiguration for Two-state changes

Input: arrivals

Output: $m(t)$

if The $(i+1)^{th}$ arrival detected at t **then**
  if $m(t-1) = m_0$ **then**
    $S_n(t) \leftarrow S_n(t-1) + T_i - \frac{1}{\lambda_0}$
    if $S_n(t) < \eta_+$ **then**
      $S_n(t) \leftarrow 0$
      $m(t) \leftarrow m(t-1) + 1$
    else
      $m(t) \leftarrow m(t-1)$
    end if
  else
    $S_n(t) \leftarrow S_n(t-1) + T_i - \frac{1}{\lambda_1}$
    if $S_n(t) > \eta_-$ **then**
      $S_n(t) \leftarrow 0$
      $m(t) = m(t-1) - 1$
    else
      $m(t) \leftarrow m(t-1)$
    end if
  end if
else
  $S_n(t) \leftarrow S_n(t-1)$
  $m(t) \leftarrow m(t-1)$
end if

An upper bound on the false alarm probability after κ arrivals given no surge happens from [12] is

$$P_{f_{ST}} \leq \frac{\kappa}{[\lambda_0 \eta_+]^2}. \tag{3.42}$$

From the above results, we know the surge/drop thresholds are important, since it not only decides when to reconfigure, but also decides the error probability. A way to assign proper thresholds is to determine by the missed detection probability and the false alarm probability, which is similar to the Neyman-Pearson test in Bayesian estimator cases.

Figure 3-7 shows the detection performances of traffic surges/drops for two Bayesian estimators $\hat{\lambda}_T(t)$ and $\hat{\lambda}_N(t)$, and a stopping-trial estimator $\hat{\lambda}_{ST}(t)$ in a single run. The similar results can be acquired from other runs. The fixed time for $\hat{\lambda}_T(t)$, the fixed count for $\hat{\lambda}_N(t)$, and the thresholds for $\hat{\lambda}_{ST}(t)$ are picked so that all three estimators' probabilities of missed detection are 1%. $T = 6.2$ for $\hat{\lambda}_T(t)$, $N = 43$ for $\hat{\lambda}_N(t)$, $\eta_- = 0.575$ for $\hat{\lambda}_{ST}(t)$. To better demonstrate the detection time and the detection accuracy, it is already assumed that the traffic rate jumps between two states. Therefore, the zig-zag shapes disappear from the plots in Fig. 3-2, Fig. 3-3, and Fig. 3-4. From Fig. 3-7, $\hat{\lambda}_{ST}(t)$ has the shortest response time to rate changes either when a traffic surge happens or a traffic drop happens. Besides, the memory reset upon the detection helps to stabilize $\hat{\lambda}_{ST}(t)$ to avoid highly frequent erroneous reconfigurations. Even if $\hat{\lambda}_{ST}(t)$ generates any false alarms, the error can be quickly corrected. $\hat{\lambda}_{ST}(t)$ requires no knowledge of *a priori* probabilities, and its detection time is shortest when the session arrival statistics provide enough confidence for reconfiguration. What is more, the algorithm is applicable beyond the Poisson traffic arrival model. As long as the inter-arrival times of traffic transactions are independent, the algorithm still reacts swiftly as the traffic rate changes, which will be discussed in the following Chapters.

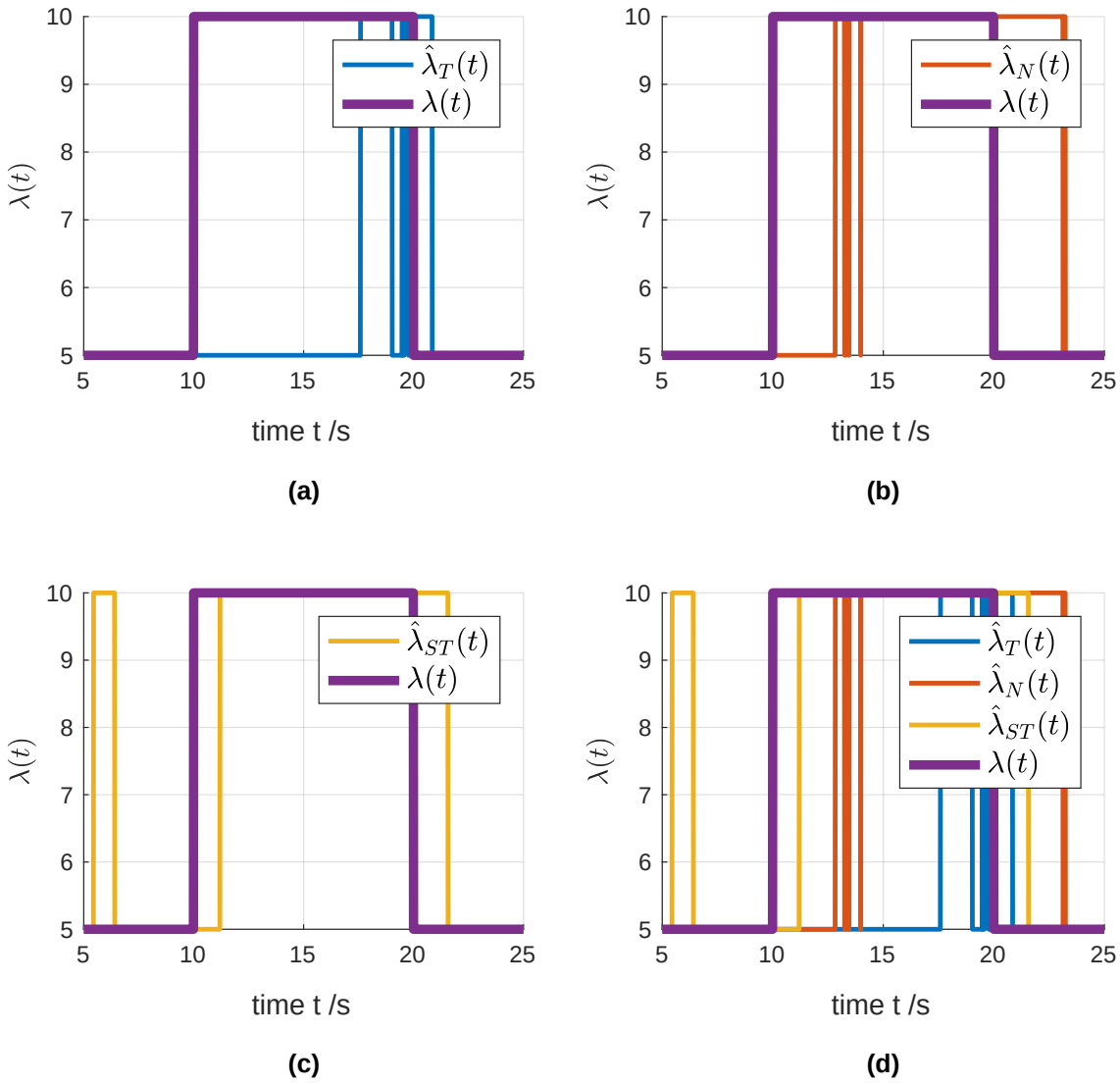

Figure 3-7: The comparison of traffic surge/drop detections among different estimators with $p_m = 1\%$. $T = 6.2$ for $\hat{\lambda}_T(t)$, $N = 43$ for $\hat{\lambda}_N(t)$, $\eta_- = 0.575$ for $\hat{\lambda}_{ST}(t)$. Here it is already assumed the traffic rate jumps between two states: (a) The fixed-time estimator $\hat{\lambda}_T(t)$; (b) The fixed-count estimator $\hat{\lambda}_N(t)$; (c) The stopping-trial estimator $\hat{\lambda}_{ST}(t)$; (d) All three estimators. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10$. [59]

3.3 Network Transient Behaviors

A good understanding of the transient behavior of the queue between each node pair is required for the design of the traffic rate change detection and network reconfiguration

algorithms. When the traffic rate or the network configuration changes, the network queueing delay also changes accordingly as one of the direct measurements of network properties. Different detection and reconfiguration schemes perform differently, and the duration of surge/drop times vary.

If no blocking happens, with the arrival rate and the service rate defined in Chapter 2, the queue between each node pair can be modeled as an $M/M/m(t)$ queue with the arrival rate $\lambda(t)$ and the service rate μ. Based on the $M/M/m(t)$ queueing model, we focus on both the peak queueing delay and the total duration of a surge/drop from the time that a surge/drop happens to the time that the network reaches a new steady state. Since the processes of traffic surge and traffic drop are quite similar, without loss of generality, we only discuss the modeling of the traffic surge process, where $\lambda(t)$ switches from λ_0 to λ_1. The modeling of the traffic drop process can be derived in a similar way and used in the final result.

As we focus on the transient behavior of the queue, the commonly used average results for the $M/M/m(t)$ queue is no longer applicable in the analysis as they are the average results over a period of time. The sudden changes after the overload cannot be reflected correctly with the average results, such as the average queueing size and the average queueing delay. Instead, we aim to find a way to model the transient queue evolution of the $M/M/m(t)$ queue to directly show the performance of different detection algorithms. A way to analytically model the transient queueing delay for the $M/M/m(t)$ queues are presented in [39]. However, the expression is too complicated and hard to use in practice. Instead, a way to approximate the continuous-time $M/M/m(t)$ queue to use a sampled-time $M/M/m(t)$ Markov chain with a very small sample unit time δ [21]. It can provide the numerical results on $M/M/m(t)$ queue. To get an analytical result, we can use the analytical results of the $M/M/1/X$ queue with the service rate $m(t)\mu$ and a large X to approximate the $M/M/m(t)$ queue, since their probability distributions are similar when the network is heavily loaded or overloaded. When network load is low, we can use the $M/M/\infty$ to approximate the performance, since almost any incoming traffic transaction will be transmitted immediately, and no queue will be built up.

A similar work discussing the average packet delay at a given node for various observation times in the time series is given in [31]. However, the results are based on the steady states of the $M/D/1$ queue and the discrete-time approximation but not the exact analytical solution. In this section, all the transient results are developed analytically.

3.3.1 Sampled-Time Markov Chain Approximation

A way to approximate the continuous-time $M/M/m(t)$ queue is using a sampled-time $M/M/m(t)$ Markov chain with a very small sample unit time δ as shown in Fig. 3-8. Every state represents the number of transactions in the system. We have $\delta \leq \frac{1}{\mu+\lambda}$ to make sure the transition probability is nonnegative. To simplify notations and avoid ambiguity, denote $\lambda(t)$ as λ in this section only. The probability of self-transitions in the sampled-time $M/M/m(t)$ Markov chain become nontrivial due to the very small unit sample time δ. It is the similar method adopted in [21] to approximate an $M/M/1$ queue with an $M/M/1$ sampled-time Markov chain. The small δ guarantees that there is at most one arrival or departure in every δ-time increment. As stated in [21], it is a relatively good approximation with the very small δ, since the probability of more than one arrival/departure, or both an arrival and a departure in a δ increment is of order δ^2 for the actual continuous-time queue system with the Poisson process.

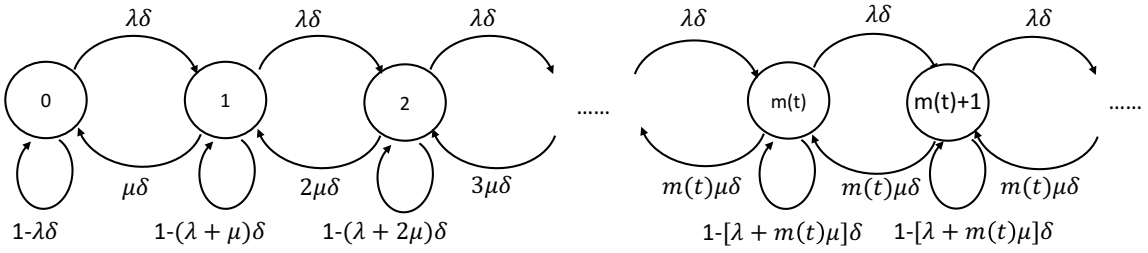

Figure 3-8: Sampled-time approximation to $M/M/m(t)$ queue with a very small time increment δ.

The stochastic matrix, which is the probability transition matrix of the $M/M/m(t)$

queue, corresponding to the Markov chain in Fig. 3-8, is

$$\begin{bmatrix}
1-\lambda\delta & \lambda\delta & 0 & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\mu\delta & 1-(\lambda+\mu)\delta & \lambda\delta & 0 & \dots & \dots & \dots & \dots & \dots & \dots \\
0 & 2\mu\delta & 1-(\lambda+2\mu)\delta & \lambda\delta & 0 & \dots & \dots & \dots & \dots & \dots \\
\vdots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \dots \\
\dots & \dots & \dots & 0 & m(t)\mu\delta & 1-(\lambda+m(t)\mu)\delta & \lambda\delta & 0 & \dots & \dots \\
\dots & \dots & \dots & \dots & 0 & m(t)\mu\delta & 1-(\lambda+m(t)\mu)\delta & \lambda\delta & 0 & \dots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots
\end{bmatrix}.$$

To facilitate analysis, we assume the whole system is finitely large with the length of X to avoid issues arising when handling infinity, and X is finitely very large. Therefore, we approximate the infinite Markov chain with a finitely large Markov chain with the largest state as state X. For state X, its transition probability to state $X-1$ is $m(t)\mu\delta$, and its transition probability to itself is $1-m(t)\mu\delta$. Therefore, we approximate the infinite transition probability matrix with a finite transition probability matrix, which is an $X \times X$ matrix with a very large X as follows: $P_{sample}=$

$$\begin{bmatrix}
1-\lambda\delta & \lambda\delta & 0 & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\mu\delta & 1-(\lambda+\mu)\delta & \lambda\delta & 0 & \dots & \dots & \dots & \dots & \dots & \dots \\
0 & 2\mu\delta & 1-(\lambda+2\mu)\delta & \lambda\delta & 0 & \dots & \dots & \vdots & \dots & \dots \\
\vdots & \ddots & \ddots & \dots & \vdots & \vdots & \vdots & \vdots & \vdots & \dots \\
\dots & \dots & \dots & 0 & m(t)\mu\delta & 1-(\lambda+m(t)\mu)\delta & \lambda\delta & 0 & \dots & \dots \\
\dots & \dots & \dots & \dots & 0 & m(t)\mu\delta & 1-(\lambda+m(t)\mu)\delta & \lambda\delta & 0 & \dots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & 0 & m(t)\mu\delta & 1-m(t)\mu\delta
\end{bmatrix}.$$

The steady-state probability of this Markov chain exists, and the distribution follows the steady-state probability of a birth-death chain. Without loss of generality, assume the queue is in a steady state initially, and assume the initial steady-state probability matrix $\Pi_{sample}(0)$ at time $t=0$ is

$$\Pi_{sample}(0) = [\pi_{0_{sample}}(0), \pi_{1_{sample}}(0), \pi_{2_{sample}}(0), ..., \pi_{M_{sample}}(0)]. \quad (3.43)$$

With the above probability transition matrix P_{sample}, we can get the transient steady-state probability of the $M/M/m(t)$ queue at time t with the initial steady-state probability matrix $\Pi_{0_{sample}}$ as

$$\begin{aligned} \Pi_{sample}(t) &= \Pi_{sample}(0) P_{sample}^{\frac{t}{\delta}} & (3.44) \\ &= [\pi_{0_{sample}}(t), \pi_{1_{sample}}(t), \pi_{2_{sample}}(t), ..., \pi_{M_{sample}}(t)]. & (3.45) \end{aligned}$$

Therefore, we can get the transient mean queue size of the $M/M/m(t)$ queue at time t as

$$Q_{sample}(t) = \sum_{i=m(t)}^{X} (i - m(t)) \pi_{i_{sample}}(t) \quad (3.46)$$

3.3.2 $M/M/1/X$ Queue Transient Behavior Approximation

The $M/M/m(t)$ queue between each node pair can be approximated by the analytical results of $M/M/1/X$ queue with the service rate $m(t)\mu$ and a large X, since their probability distributions are similar when the network is highly loaded or overloaded. When the network load is low, we can use the $M/M/\infty$ queue to approximate the probability distribution. Since almost no queue will be built up in the low load situation, the network performance is good in terms of no queueing delay. Therefore, there is no need to discuss the transient queue behavior in the low load, and we only focus on the $M/M/1/X$ queue transient behavior approximation.

The transient behavior of the $M/M/1/X$ queue can be developed from the real-time probability distribution of the number of traffic transactions in the system. Denote $p_n^k(t)$ as the probability that n traffic transactions are in the system at the

current time t given k transactions in the system at $t = 0$. Assume X as the maximum number of transactions in the system, where X is very large to approximate the system with the infinite buffer. From the analytical results of transient behavior of $M/M/1/X$ queues in [37], the corresponding time-dependent solution to $p_n^k(t)$ is

$$p_n^k(t) = \pi_n + \frac{2\rho^{\frac{1}{2}(n-k)}}{X+1} \sum_{i=1}^{X} \left(\frac{\mu}{\beta_i}\right) \cdot \left[\sin\frac{ik\pi}{X+1} - \sqrt{\rho}\sin\frac{i(k+1)\pi}{X+1}\right] \cdot \cdot \left[\sin\frac{in\pi}{X+1} - \sqrt{\rho}\sin\frac{i(n+1)\pi}{X+1}\right] e^{-\beta_i t}. \quad (3.47)$$

where $\rho = \frac{\lambda(t)}{\mu}$, and π_n is the steady-state probability for state n in the $M/M/1/X$ queue in [37] as

$$\pi_n = p_n^k(\infty) = \frac{1-\rho}{1-\rho^{X+1}}\rho^n \qquad n = 0, 1, ..., X. \quad (3.48)$$

The expression for β_i in [37] is

$$\beta_i = \lambda(t) + \mu - 2\sqrt{\lambda(t)\mu}\cos\left(\frac{i\pi}{X+1}\right), \quad (3.49)$$

where $i = 1, 2, ..., X$, and $n = 0, 1, 2, ..., X$.

The mean queue length at time t in [37] is

$$Q(t) = \sum_{n=1}^{X}(n-1)p_n^k(t). \quad (3.50)$$

3.3.3 Transient Behavior of the Queue

Figure 3-9 shows both the analytical $M/M/1/X$ queue approximation and the simulated transient behaviors of the queue for a traffic surge followed by a proper reconfiguration. As shown in Fig. 3-9, when a traffic surge happens, the average queue

size grows sharply as the change has not been detected so that the system is still in the previous configuration with insufficient wavelengths. Once the estimator detects the surge, it will report it to the system, and the system will allocate more wavelengths to digest the surging queue. When the reconfiguration is finished, the queue in the system begins to diminish and finally reaches a new steady state with little or no queueing delay. The simulated transient behavior is generated by simulating the arrival process of the traffic and directly recording the transient queue size. The analytical result and the simulated result agree with each other, and it shows that the $M/M/1/X$ queue with the service rate $m(t)\mu$ and a large X is a good approximation of the $M/M/m(t)$ queue analytically. We will adopt it in our analysis.

To better quantify the transient behavior of the queue, we describe the evolution of the queue after a surge with several critical variables as shown in Fig. 3-9. Define t_1 as the time that a traffic surge happens, where the traffic arrival rate switches from λ_0 to λ_1. t_2 is the time that the change is detected and the network is reconfigured. We assume that a reconfiguration is completed instantaneously when the surge is detected to facilitate the analysis. Even if a certain delay happens between when surge is detected and when the network is reconfigured, the delay is short and usually constant compared to the time to detect the surge. It can be considered in addition to the time it takes to detect the surge. t_3 is the time that the network with the new wavelength assignment (*i.e.*, the new service rate) reaches the steady state. The detection time is defined as $\tau_1 = t_2 - t_1$. The queue settling time is defined as $\tau_2 = t_3 - t_2$. The duration of a surge is defined as

$$\tau_{surge} = \tau_1 + \tau_2, \tag{3.51}$$

where we assume delays other than τ_1 and τ_2 can be ignored.

Since the average peak queue size Q_{peak} is reached at t_2, the average peak queueing delay Γ_{peak} is also reached at t_2. Obviously, both the average peak queueing delay Γ_{peak} and τ_2 depend on τ_1. A long τ_1 can result in a large Q_{peak} (and a large Γ_{peak}) and

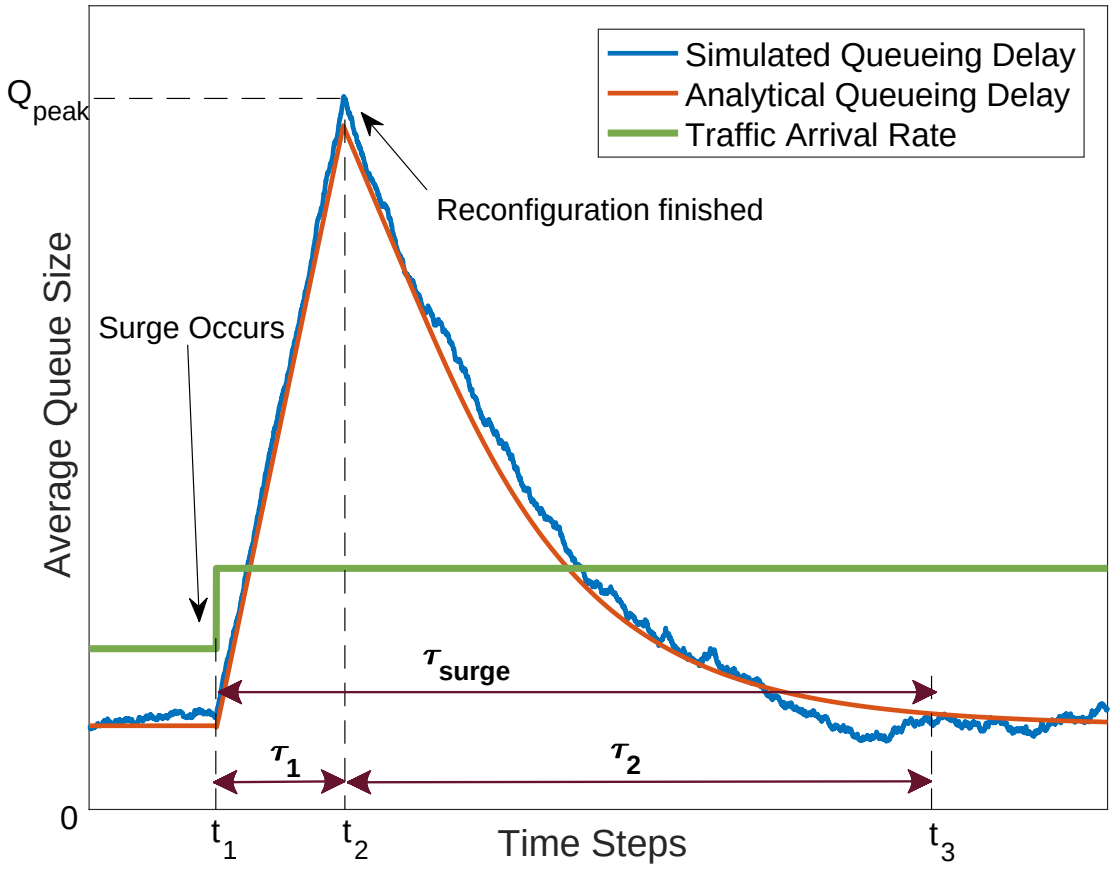

Figure 3-9: (a) Simulated and analytical results of the evolution of queue size for a network surge followed by a proper reconfiguration for an $M/M/1/X$ queue. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10, \mu = 6$.[59]

a long τ_2 if the mismatch between the traffic surge and the existing configuration is large. What is worse, it also means more resources need to be allocated immediately to digest the transiently large Q_{peak} and the transiently long τ_2, which increases the network management and control burdens. Therefore, an estimator that can quickly respond to changes will lead to both a shorter peak queueing delay and a shorter queue settling time, which is desired. The modeling of τ_1, τ_2, and Γ_{peak} will be discussed in the following sections.

Figure 3-10 shows the comparison between the $M/M/1/X$ queue with the service rate $m(t)\mu$ and a large X approximation and the sampled-time $M/M/m(t)$ Markov

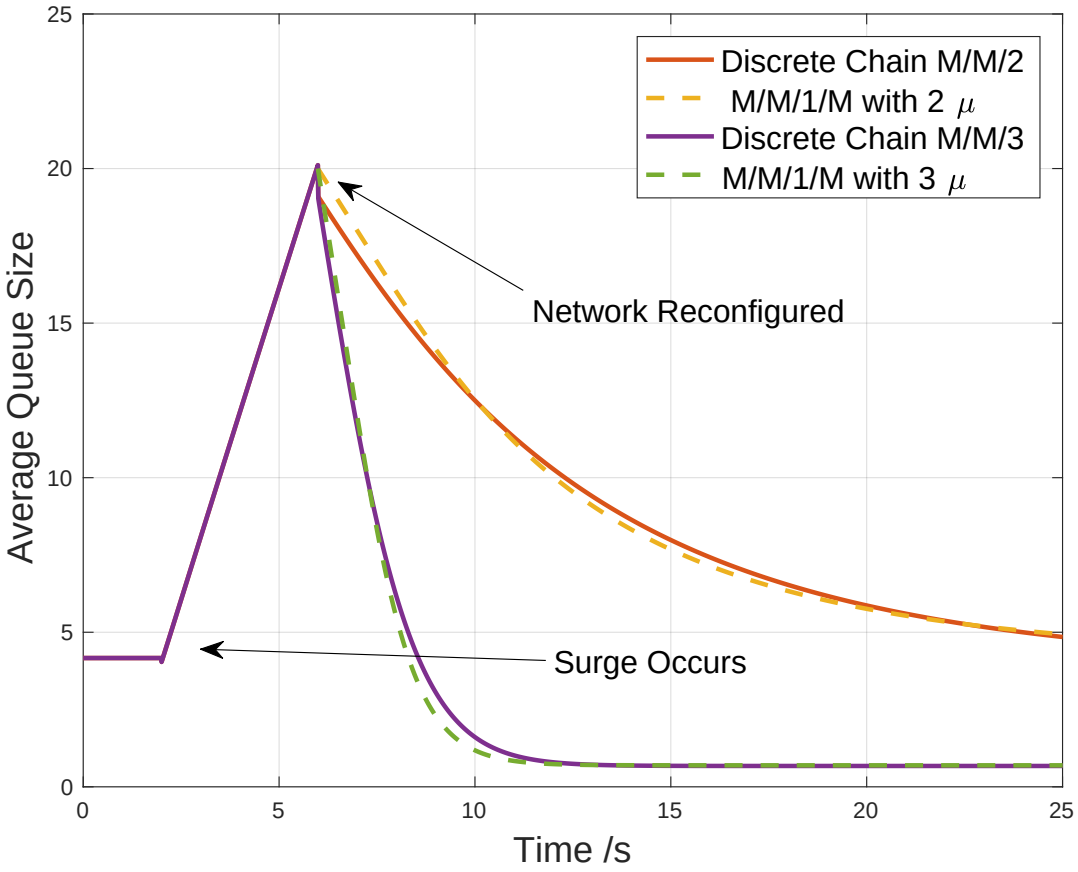

Figure 3-10: The transient behavior of the average queue size when a surge occurs and a proper reconfiguration is made later. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10, \mu = 6$. [59]

chain. Given the $M/M/1/X$ queue with the service rate $m(t)\mu$ and a large X is good approximation with the simulation as shown in Fig. 3-9, the agreement of the approximation results shows that the sampled-time $M/M/m(t)$ Markov chain is a good numerical approximation. Two approximations agree with each other with different $m(t)\mu$. Besides, we find that the queue can be quickly digested and τ_2 is short if more wavelengths are assigned after the surge. Since the analytical results of the $M/M/1$ queue with the service rate $m(t)\mu$ and a large X can successfully approximate the transition behavior of the $M/M/m(t)$ queue with different $m(t)$, we

will use the approximation in the following sections.

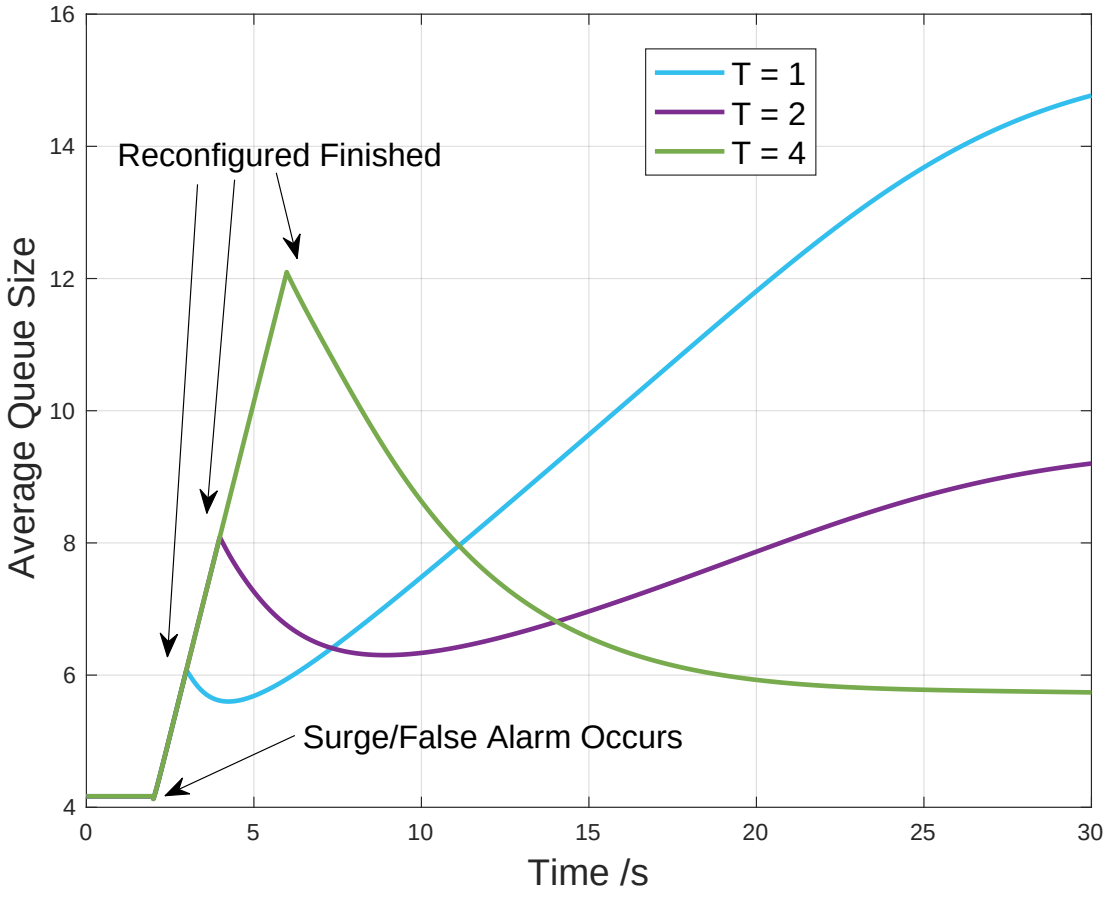

Figure 3-11: Transient behavior of the average queue size with only one reconfiguration for different detection times of the fixed-time estimator $\hat{\lambda}_T(t)$. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10, \mu = 6$.

The transition of average queue size with one reconfiguration for different detection times, averaging both detections and false alarms is shown in Fig. 3-11. Here the fixed-time estimator $\hat{\lambda}_T(t)$ is used to show the results of different detection time lengths. The same conclusion can be drawn from the results of either the fixed-count estimator or the stopping-trial estimator. A long detection time τ_1 will incur a long queue settling time τ_2 and a large average queue size Q_{peak}. The curves diverge after the peak in Fig. 3-11 because the probability of missed detection and false alarm

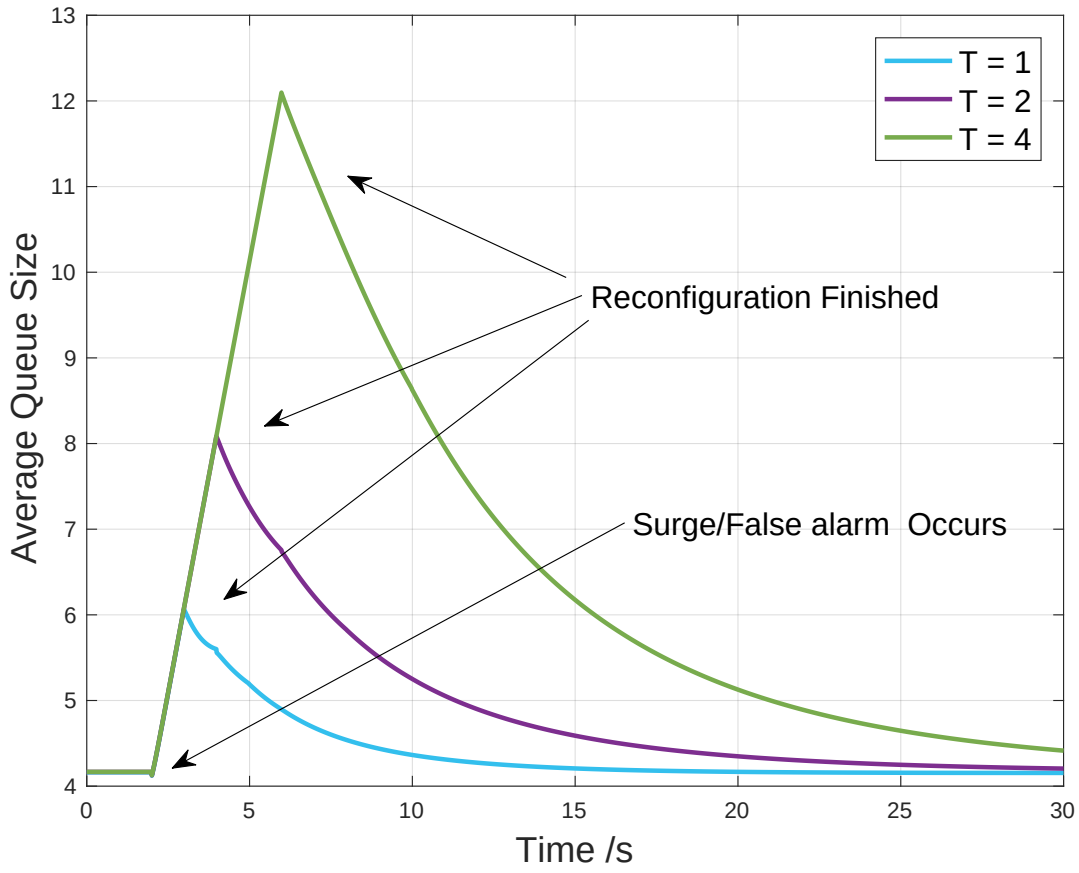

Figure 3-12: Transient behavior of the average queue size with multiple reconfigurations for different detection times of the fixed-time estimator $\hat{\lambda}_T(t)$. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10, \mu = 6$.

differ. Since no further correction is allowed in Fig. 3-11, a missed detection can happen and lead to the continuing increase of the queue. Though a short detection time can lead to a fast reconfiguration, it also suffers from a high missed detection probability so that the average queue size keeps increasing and makes the network unstable. A long detection time incurs a high peak queueing delay in exchange for a low missed detection probability, which also burdens the network control. Therefore, a short detection time of network rate changes is one of the keys in designing the cognitive control of wavelength assignment. Figures with shapes similar to the average

packet delay for various observation times in the time series can be found in [31]. However, the results are based on the steady states of the $M/D/1$ queue and the discrete-time approximation but not the exact analytical solution.

The transition of average queue size with more than one reconfiguration allowed for different detection times averaging both detections and false alarms are shown in and Fig. 3-12. Again, the fixed-time estimator $\hat{\lambda}_T(t)$ is used to show the results of different detection time lengths. The same conclusion can be drawn from the results of either the fixed-count estimator or the stopping-trial estimator. Similarly as shown in Fig. 3-11, a long detection time τ_1 will incur a long queue settling time τ_2 and a large average queue size Q_{peak}. Continuing assessment with multiple reconfigurations enabled can help to reduce the peak queue delay as shown in Fig. 3-12, where the slightly zig-zag shapes are the results of correcting errors at different times. Continuing reconfigurations compensate for the detection inaccuracy of the estimators and should be enabled to increase the cognitive control performance.

3.3.4 Detection Time τ_1

Detection time τ_1 is the time that the estimator needs to detect the traffic change. It is crucial in the network delay transition, since it decides both the queue settling time τ_2 and the peak queueing delay Γ_{peak}. In this section, we develop the detection time τ_1 for different estimators developed in the previous sections. The value of τ_1 depends on both the base detection duration and the detection accuracy. For example, without error, $\hat{\lambda}_T(t)$ needs at least the length of the fixed observation interval T to detect a traffic change, *i.e.*, $\tau_1 = T$. Similarly, $\hat{\lambda}_N(t)$ needs at least a length of counting a fixed number of arrivals N to detect the traffic change, *i.e.*, $\tau_1 = \frac{N}{\lambda(t)}$. The actual length varies depending on the detection accuracy. For example, a missed detection will increase the detection time τ_1.

Fixed-time estimator $\hat{\lambda}_T(t)$

$\lambda_T(t)$ has a base detection duration T. If no rate change is detected within T, the algorithm keeps working continuously until it catches a change. Let ΔT be the time it takes the algorithm to detect a change after the miss within T. We can model ΔT as a random walk where an arrival comes or leaves in a short unit time δ.

In the event of missed detection, denote η as the threshold of determining the probability of detection p_{d_T} and missed detection $p_{m_T} = 1 - p_{d_T}$. When a surge occurs, we have the average detection of $\lambda_T(t)$ as

$$\tau_{1_T} = p_{d_T} T + (1 - p_{d_T})(T + \Delta T), \tag{3.52}$$

where ΔT is the average delay beyond the base detection time T modeled into a random walk when finally the detection is found. We have ΔT as

$$\Delta T = \sum_{n=0}^{\lfloor \eta - 1 \rfloor} P(n) \frac{(\eta - n)^2}{2\lambda_1 \delta (1 - \lambda_1 \delta)}, \tag{3.53}$$

where $P(n)$ follows a Poisson distribution with λ_1.

In the event of false alarms, we have the average detection of $\lambda_T(t)$ as

$$\tau_{1_T} = (1 - p_{f_T}) T + p_{f_T}(T + \Delta T), \tag{3.54}$$

where the ΔT is

$$\Delta T = \sum_{n=\lfloor \eta \rfloor}^{\infty} P(n) \frac{(n - \eta)^2}{2\lambda_0 \delta (1 - \lambda_0 \delta)}, \tag{3.55}$$

and where $P(n)$ follows a Poisson distribution with λ_0.

Fixed-count estimator $\hat{\lambda}_N(t)$

$\lambda_N(t)$ needs a duration of N arrivals as the detection time. The results will be updated with new arrivals. In the event of missed detection, denote the probability of detection p_{d_N} and missed detection $p_{m_N} = 1 - p_{d_N}$. The chance to catch the change in an arrival is p_{d_N} and the chance to miss it is p_{m_N}. If missed, the detection needs to wait until the next arrival, and the situation recurs until the change is caught. We can formulate the average detection time when a surge occurs as

$$\tau_{1_N} = \sum_{n=1}^{\infty} p_{d_N}(1 - p_{d_N})^{n-1} \frac{N + n - 1}{\lambda_1}. \tag{3.56}$$

In the event of false alarms, denote the probability of false alarm as p_{f_N}. Similarly, we have

$$\tau_{1_N} = \sum_{n=1}^{\infty} (1 - p_{f_N}) p_{f_N}^{n-1} \frac{N + n - 1}{\lambda_0}. \tag{3.57}$$

Stopping-trial estimator $\hat{\lambda}_{ST}(t)$

Different from the fixed-time estimator $\hat{\lambda}_T(t)$ and the fixed-count estimator $\hat{\lambda}_N(t)$ with the pre-fixed base detection time, the stopping-trial estimator $\lambda_{ST}(t)$'s detection time is totally based on the traffic arrivals. As discussed in [12], if the traffic comes frequently, the detection time of $\lambda_{ST}(t)$ will be short; if the traffic comes at a slow rate, the detection time of $\lambda_{ST}(t)$ will be long. The thresholds (determined by the error probability) to add/drop wavelength also decides the detection time [12].

When a traffic surge occurs, $\lambda(t)$ switches from the non-surging state λ_0 to the surging state λ_1. $\lambda_{ST}(t)$ needs the average stopping time $E[T_i]E[J]$ to make the decision, where J is the time in the unit of arrivals that a threshold is crossed [12]. $E[J]$ is the average number of arrivals that a threshold is crossed, and $E[T_i]$ is the average inter-arrival time. Based on [12], $E[J]$ could be derived from Wald's equality

as

$$E[S_J] = E[T_i - \frac{1}{\lambda_0}]E[J] \tag{3.58}$$

$$\Leftrightarrow E[J] = \frac{E[S_J]}{E[T_i - \frac{1}{\lambda_0}]}, \tag{3.59}$$

where $E[T_i - \frac{1}{\lambda_0}]$ is the expected step size and the initial state is in λ_0 before a surge happens. Notice that $E[T_i - \frac{1}{\lambda_0}] \neq 0$, as $E[T_i] = \frac{1}{\lambda_1}$.

Based on [12], with $E[T_i] = \frac{1}{\lambda_1}, E[S_J] = \eta_+$, the average detection time when a surge occurs is

$$\tau_{1_{ST}} = E[T_i]E[J] = \frac{E[T_i]E[S_J]}{E[T_i - \frac{1}{\lambda_0}]} = \frac{\lambda_0\eta_+}{\lambda_0 - \lambda_1}. \tag{3.60}$$

In the event of false alarms, $E[T_i] = \frac{1}{\lambda_0}$ as the traffic is still in the non-surging state λ_0. $E[T_i - \frac{1}{\lambda_0}] = 0$, which makes Wald's equality inapplicable as the denominator cannot be zero. In this case, we can instead use the second derivative of Wald's identity as

$$E[S_J^2] = E[J]\sigma^2_{(T_i - \lambda_0)} \tag{3.61}$$

$$\Leftrightarrow E[J] = \frac{E[S_J^2]}{\sigma^2_{(T_i - \lambda_0)}}, \tag{3.62}$$

where $\sigma^2_{(T_i - \lambda_0)}$ is the variance of the variable $(T_i - \lambda_0)$, which equals the variance of T_i in the non-surging state.

With $E[T_i] = \frac{1}{\lambda_0}, \sigma^2_{(T_i - \lambda_0)} = \frac{1}{\lambda_0^2}, E[S_J^2] = \eta_+^2$, we have

$$\tau_{1_{ST}} = E[T_i]E[J] = \frac{E[T_i]E[S_J^2]}{\sigma^2_{(T_i - \lambda_0)}} = \lambda_0\eta_+^2. \tag{3.63}$$

3.3.5 Peak Queueing Delay Γ_{peak}

The peak queueing delay Γ_{peak} is an important measure of the network, as it indicates the worst case of delay if a mismatch happens between the traffic pattern and the configuration. Also, a large Γ_{peak} requires more resources, such as wavelengths, to go to a new steady state quickly.

We can develop the average queueing delay from the average queue size. For the $M/M/1$ queue, the queueing delay of the incoming transaction with $(n+1)$ transactions already in the system (*i.e.*, n in the queue) is the total transmission time of all $(n+1)$ transactions. Given the average transmission delay per transaction as $\frac{1}{\mu}$ and the average queue length $Q(t)$, the average queueing delay for the incoming transaction at time t is

$$\Gamma_q(t) = \frac{Q(t)+1}{\mu} = \frac{[\sum_{n=1}^{N}(n-1)p_n^m(t)]+1}{\mu} \tag{3.64}$$

where the full expression of $p_n^m(t)$ is shown in Equation 3.47.

Though Little's Law is a common way to derive the average queueing delay with the queue size and the average effective arrival rate [32], we will not adopt Little's Law in this work since it depicts the long-term average queue size in a stationary system rather than the transient behaviors of the queue [3]. Our detection algorithm has to react to the transient behavior of the system to catch any traffic change as soon as possible.

For the $M/M/m(t)$ queue in our analysis, we can prove the following proposition on its queueing delay.

Proposition 1 The queueing delay distribution for the $(n+1)^{th}$ transaction in the $M/M/m(t)$ queue with the service rate μ is the same with that of the $M/M/1$ queue with the service rate of $m(t)\mu$.

Proof. For the $M/M/1$ system with the service rate of $m(t)\mu$, a transaction needs to wait if the wavelength is occupied. Since the inter-arrival time is exponentially distributed, the probability that an occupied wavelength is still busy with the current

transmission by time T is $e^{-m(t)\mu T}$. The probability that the current transmission is finished by time T is $1 - e^{-m(t)\mu T}$. For the $(i+1)^{th}$ transaction in the queue, the system must finish the transmission i times in order that the $(i+1)^{th}$ transaction in the queue start to be transmitted. Thus the result is to take the $i-$fold convolution of $1 - e^{-m(t)\mu T}$ [39].

Similarly, for a $M/M/m(t)$ system with a service rate of μ, the probability that an occupied wavelength is still busy with the current transmission by time T is $e^{-\mu T}$. The probability that a wavelength channel will complete the transmission of the current transaction by time T is $1-e^{-\mu T}$. Given $m(t)$ wavelengths in the system, the probability that all $m(t)$ wavelengths continue to be busy with the same transactions by time T is $e^{-m(t)\mu T}$, and the probability that any of the channels becomes available is $1 - e^{-m(t)\mu T}$. For the $(i+1)^{th}$ transaction in the queue, the system must finish the transmission i times in order that the $(i+1)^{th}$ transaction in the queue can be transmitted, and similarly the result is $i-$fold convolution of $1-e^{-m(t)\mu T}$, which is the same waiting time distribution of the $M/M/1$ system with the service rate of $m(t)\mu$.

Therefore, we prove that the distribution of waiting time in the queue for the $(i+1)^{th}$ transaction in the queue is the same for both queue systems. We can then prove the following proposition.

Corollary of Proposition 1 The average queueing delay for an $M/M/m(t)$ queue with the service rate μ is the same as that of an $M/M/1$ queue with the service rate of $m(t)\mu$.

Proof. The average queueing delay is the weight averaging of the queueing delay distribution over time. Given the queueing delay distributions are the same for both queue systems from **Proposition 1**, the average queueing delay of $M/M/m(t)$ queue with the service rate μ and the $M/M/1$ queue with a service rate of $m(t)\mu$ are the same.

We can develop the peak average queueing delay Γ_{peak} from the average queue size Q_{peak} with the above analysis. Γ_{peak} is reached at time t_2, when the traffic surge is detected and the network is reconfigured. Therefore, we have the peak average queueing delay Γ_{peak} for an $M/M/m(t)$ queue as

$$\Gamma_{peak} = \Gamma_q(t_2) = \frac{Q_{peak} + 1}{m(t)\mu}. \tag{3.65}$$

Figure 3-13 shows the normalized average peak delays versus surge changes and detection times τ_1. The normalized average peak delay is the peak queueing delay normalized by the average transmission delay $\tau_{trans} = \frac{L}{R}$. The normalized peak delay increases with the increase of detection time τ_1. Therefore, a fast response time can help to avoid severe peak queueing delays. On the other hand, a larger surge traffic rate yields a longer normalized delay, where the surge traffic rate is the new arrival rate. A larger new arrival rate will build up the queue faster and thus have a longer queueing delay.

The transients of the normalized average queueing delays with one reconfiguration for different detection times τ_1 averaged over both detections and false alarms are shown in Fig. 3-14. The performance of different τ_1 diverge after the peak in Fig. 3-14 because p_m and p_f differ. Though a short detection time can lead to a low Γ_{peak}, it also suffers from a high missed detection probability so that the average queue size keeps increasing, making the network unstable. A long detection time incurs a high Γ_{peak} in exchange for a low missed detection probability, and increases delays and degrades users' quality of service, which is not preferred. A similar work discussing the average packet delay in the steady state of $M/D/1$ queue at a given node for various observation times based on the discrete-time approximation but not on the analytical solution is given in [31].

Therefore, an optimized detection time is important in designing the cognitive control of wavelength assignment as it results in a low peak queueing delay if the detection algorithms without continuous assessments are used. However, the trade-off between the detection time and the detection error requires an improvement in the use of continuous assessments with multiple reconfigurations enabled.

The transients of the normalized average queueing delays with more reconfigurations for different detection times τ_1 averaged over both detections and false alarms

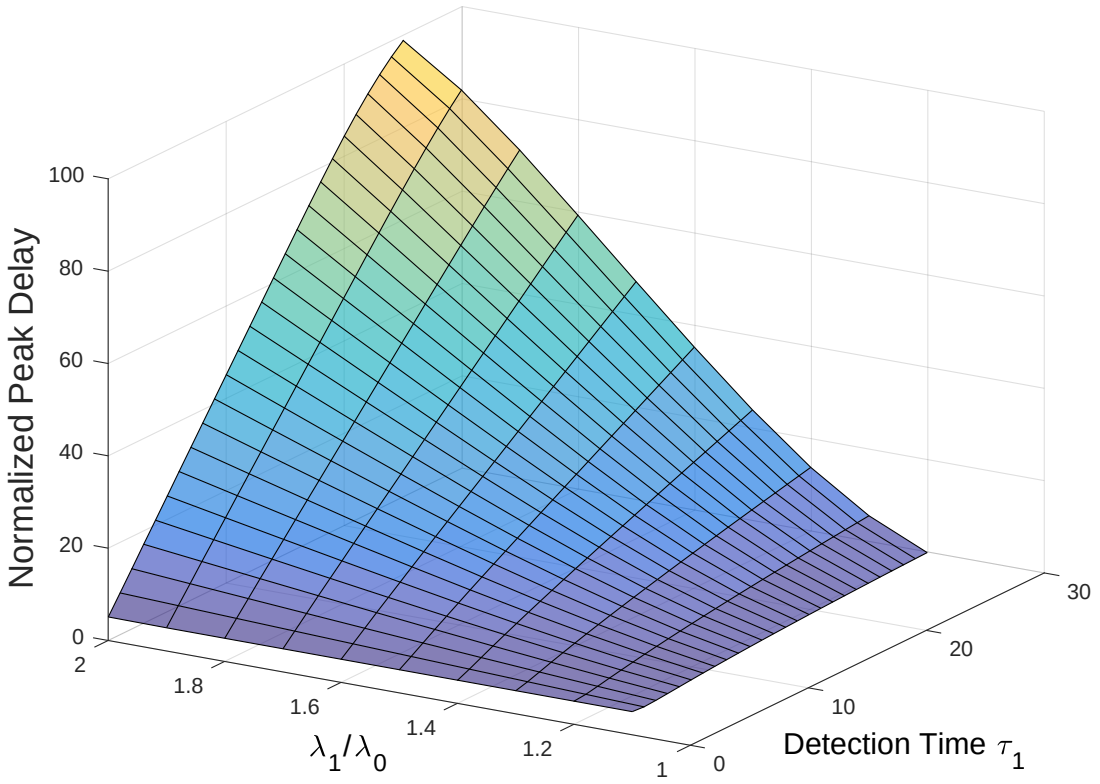

Figure 3-13: Normalized average peak delays versus surge changes and detection times τ_1. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \mu = 6$. [59]

are shown in Fig. 3-15. As expected, an algorithm with continuous assessments and reconfigurations will reduce the peak queueing delay as shown in Fig. 3-15. Even if an error occurs in the first reconfiguration, the continuous assessments enable the system to correct the errors in a timely manner. The zig-zag shapes come from correcting errors previously made. Hence, continuous assessments and reconfigurations algorithms can compensate for estimators' detection inaccuracy and should be used.

3.3.6 Queue settling time τ_2

τ_2 is the time that the network needs to serve the traffic transactions accumulated in the queue up to τ_1 and settles to the steady state with the new service rate. We have

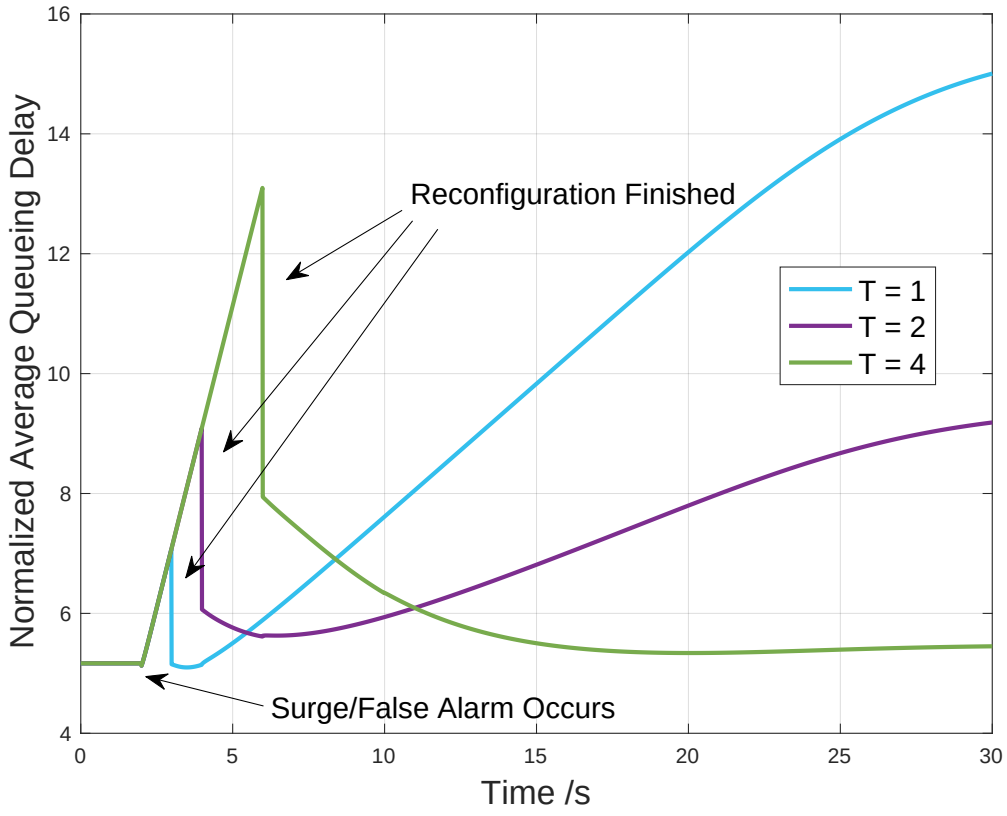

Figure 3-14: Average normalized queueing delay for different detection times τ_1 with one reconfiguration for different detection times of the fixed-time estimator $\hat{\lambda}_T(t)$. The results are the average of both detections and false alarms. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10, \mu = 6$.[59]

$$\tau_2 = t_3 - t_2, \tag{3.66}$$

and where t_3 is

$$t_3 = \min_{t>t_2} t, \;\; s.t. \; \Gamma_q(t) = \Gamma_{steady}, \tag{3.67}$$

where Γ_{steady} is the new steady state queueing delay after a proper reconfiguration.

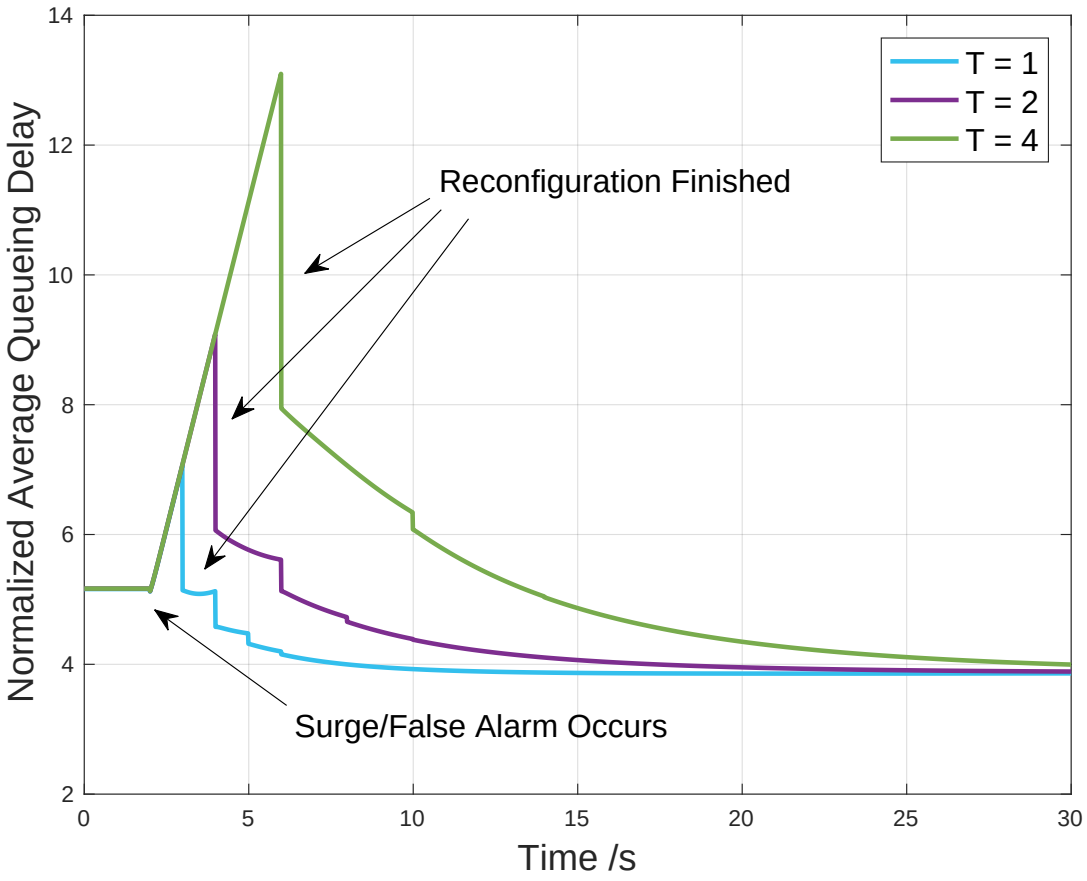

Figure 3-15: Average normalized queueing delay for different detection times τ_1 with multiple reconfigurations for different detection times of the fixed-time estimator $\hat{\lambda}_T(t)$. The results are the average of both detections and false alarms. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10, \mu = 6$.[59]

We can find a bound on τ_2 using the convergence properties of the sampled-time $M/M/m(t)$ Markov chain, and it will be discussed detailedly in Chapter 5.

3.4 Cost-Driven Network Reconfiguration Scheme

Network reconfigurations are made not only based on performance, but on network costs. There is a trade-off between the queueing delay and the cost of a wavelength in determining the reconfiguration algorithm. More wavelengths can bring a lower

queueing delay but come at a higher total cost. The cost of a wavelength includes both the capital expenditure of fiber, switches, and amplifiers, and the operating expenditure of setting up wavelengths. Sometimes, it is acceptable to make a wrong decision as long as the incurred total cost is low. On the other hand, we may not want to add a new wavelength for a transient traffic surge, since it may cost much more to reconfigure the network than to tolerate the transient delay increase.

In this section, we will add the network cost into consideration when making the reconfiguration. Also, the addition or subtraction of multiple wavelengths is discussed.

3.4.1 Network Operating Cost Model

The network operating cost model should include both the cost of delay and the cost of wavelengths. Denote the cost parameter C_d as the cost per unit of the normalized queueing delay, and denote the cost parameter C_w as the cost per wavelength. With total $m(t)$ wavelengths assigned at time t and the transient queueing delay $\Gamma_q(t)$ at time t, the total cost is

$$C_{total} = C_w m(t) + C_d \Gamma_q(t). \tag{3.68}$$

where the full expression of $\Gamma_q(t)$ is shown in Equation 3.65.

Figure 3-16 shows the transient behaviors of the total costs of the different algorithms if only one reconfiguration is allowed, averaging with detections and false alarms. $C_w = C_d = 100$, and the target probability of missed detection for all three estimators is 10%. A short detection time will lead to a lower peak total cost. However, even though the estimators respond differently to the surge, the single decision nature of this particular case with no further correction upon erroneous actions will lead all of them to higher costs eventually. It is driven by high queueing delay due to missed detection errors. Therefore, continuous assessment with multiple reconfigurations is encouraged.

Figure 3-17 show the transient behaviors of the total costs of the different algo-

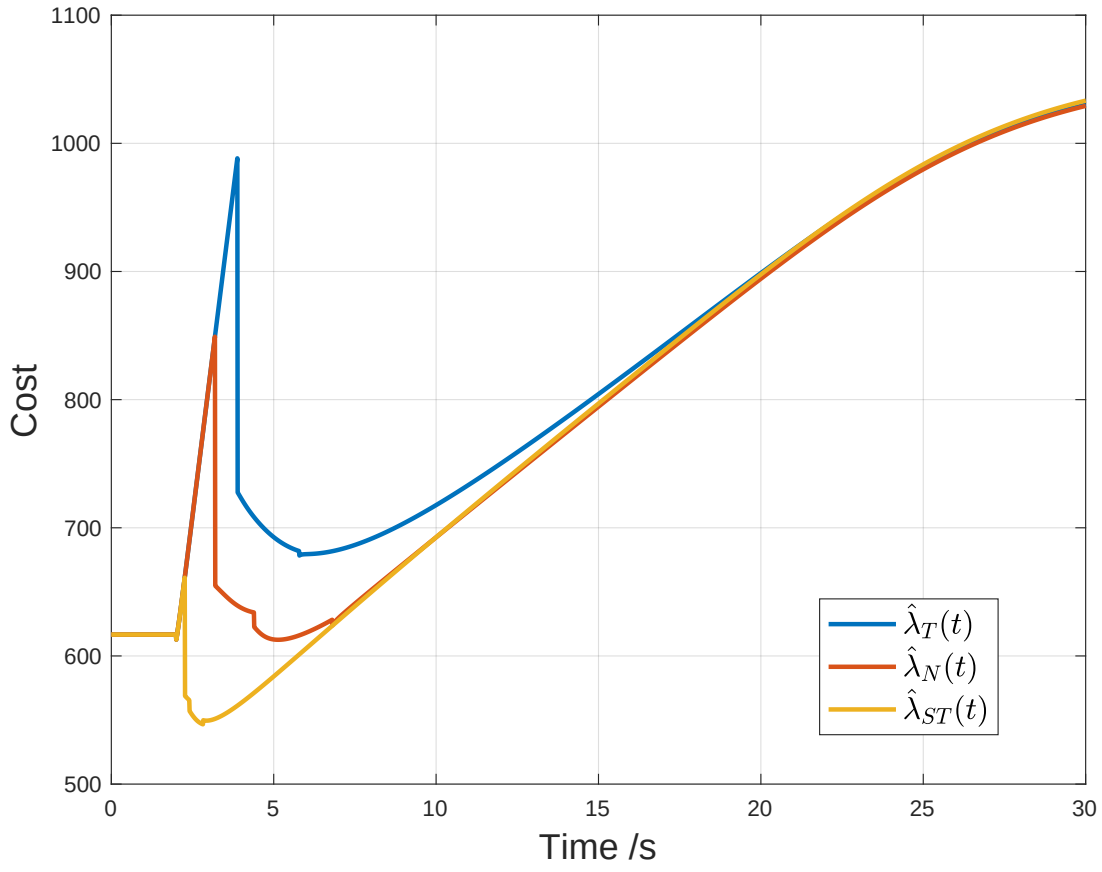

Figure 3-16: Cost transition versus time with one reconfiguration. The results are the average of both detections and false alarms. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10, \mu = 6$. $C_w = C_d = 100$, $p_m = 10\%$. [59]

rithms if multiple reconfigurations are allowed, averaging with detections and false alarms. $C_w = C_d = 100$, and the target probability of missed detection for all three estimators is 10%. When continuous assessment with multiple reconfigurations is enabled, the system can effectively correct errors and bring down costs. Even though missed detections/false alarms have occurred, the continuous reconfigurations are able to correct the errors in a timely manner. Due to its fast response to changes, the stopping-trial estimator requires the shortest time to reconfigure correctly, and yields the lowest total cost. Therefore, we recommend the stopping-trial estimator $\lambda_{ST}(t)$ for traffic detection in terms of the low network operating cost.

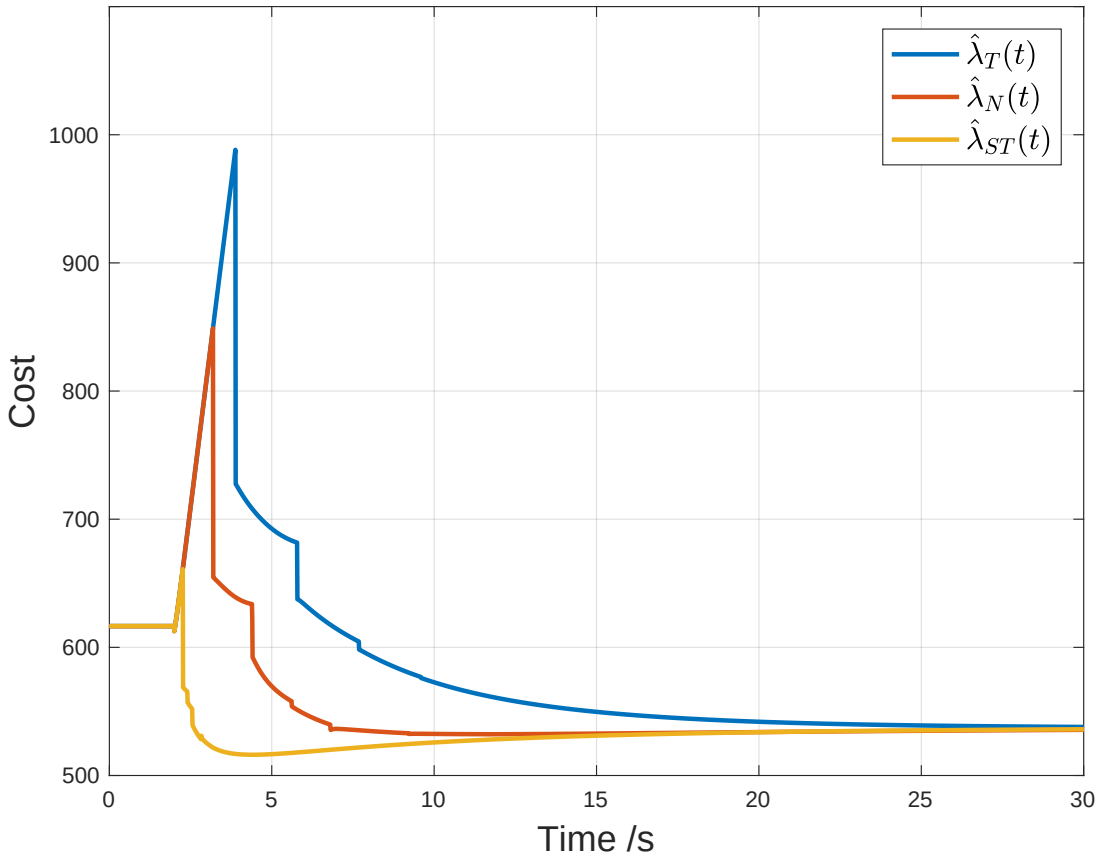

Figure 3-17: Cost transition versus time with multiple reconfigurations. The results are the average of both detections and false alarms. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10, \mu = 6$. $C_w = C_d = 100$, $p_m = 10\%$. [59]

3.4.2 Multiple Wavelengths Addition and Subtraction

A high peak queueing delay Γ_{peak} requires more resources to digest the cumulated queue in a certain rate after the surge.The addition and subtraction of multiple wavelengths can be used to deal with severe traffic surges, since more additional wavelengths assigned at once can better reduce the queueing delay. Considering the trade-off between the queueing delay and the cost of any additional wavelength, we need to find an optimal combination of both factors to achieve the optimal total cost. Figure 3-18 shows the cost comparisons of different estimators with different number

of wavelengths assigned. The queueing delay is also decided by the number of wavelengths as in Equation 3.65. The stopping-trial estimator requests a smaller number of wavelengths, realizing higher cost efficiency than the other two estimators, since its fast response helps to avoid a high peak queueing delay.

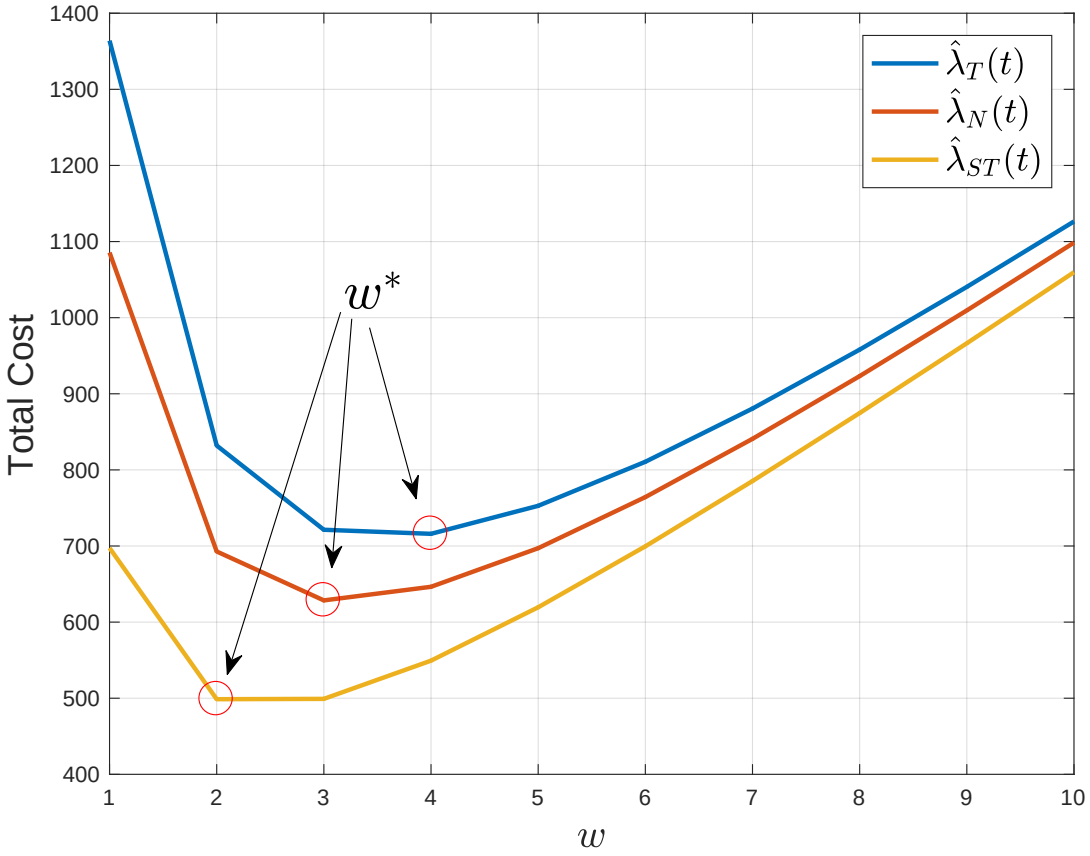

Figure 3-18: Cost comparison of different estimators with $p_m = 10\%$. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10, \mu = 6$. $C_w = C_d = 100$. [59]

When the assignment of multiple wavelengths is allowed, we further modify the cost model, and use w to denote the number of wavelengths that should be used. The total cost is

$$C_{total} = C_w w + C_d \Gamma_{peak} \frac{w_0}{w}, \tag{3.69}$$

where w_0 is the number of wavelength assigned before the surge occurs.

It is possible find the optimal number of wavelengths to assign, given the total cost constraints. To optimize it, take the derivative on the total cost as

$$\begin{aligned} \frac{dC_{total}}{dw} &= 0 && (3.70) \\ \Leftrightarrow \frac{d(C_w w + C_d \Gamma_{peak} \frac{w_0}{w})}{dw} &= 0 && (3.71) \\ \Leftrightarrow C_w - \frac{C_d \Gamma_p eak}{w^2} &= 0. && (3.72) \end{aligned}$$

Then we have the optimal number of wavelengths w^* as

$$w^* = \left\lceil \sqrt{\frac{C_d \Gamma_{peak} w_0}{C_w}} \right\rceil, \tag{3.73}$$

where the ceiling used as the number of wavelengths is an integer.

Figure 3-19 shows the optimal number of wavelengths comparison versus the cost parameter ratio C_w/C_d of different estimators. The stopping-trial estimator uses the smallest number of wavelengths for the same quality of service among all three estimators for the different combinations of cost parameters. This is because the stopping-trial estimator's fast reaction to traffic changes helps save costs by reducing the queueing delay, and therefore relaxes the demand on the number of wavelength increments during surges.

Figure. 3-20 shows the optimal number of wavelength comparisons versus load ρ of different estimators. The stopping-trial estimator uses the smallest number of wavelengths for the same quality of service among all three estimators for different network loads. The stopping-trial estimator is recommended as the algorithm for reconfigurations.

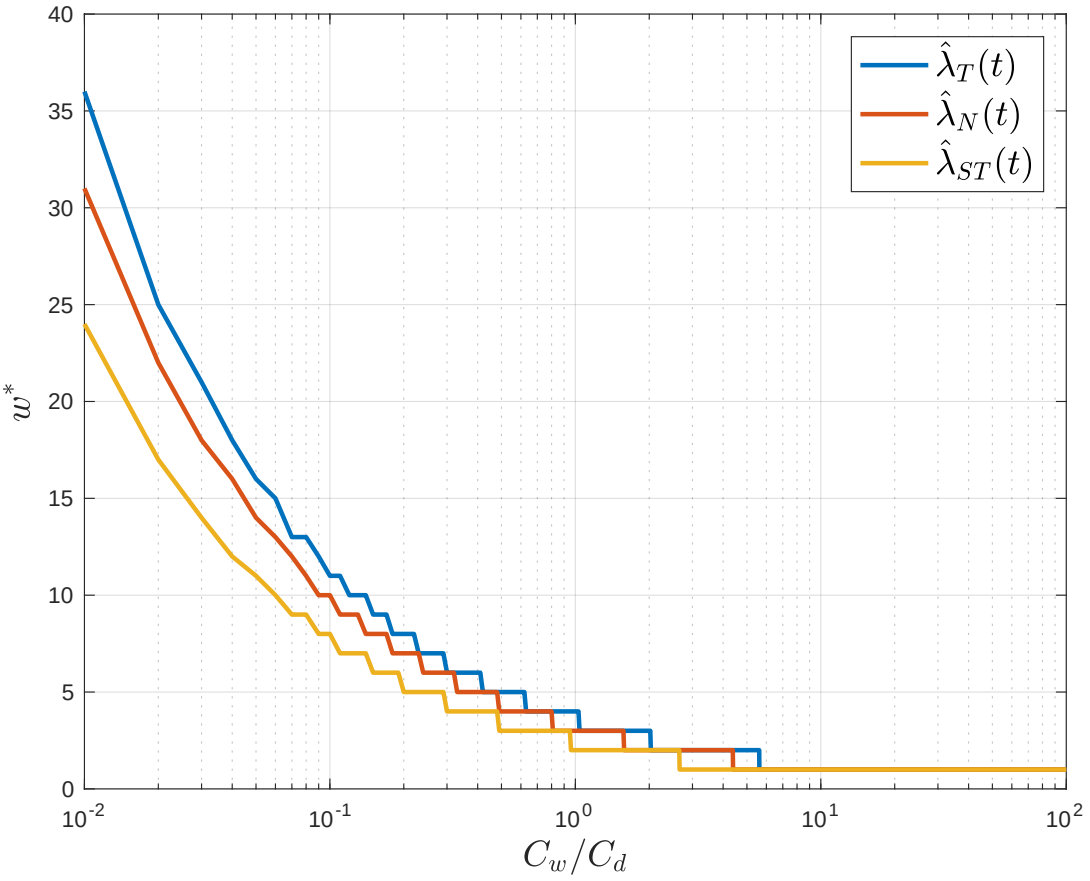

Figure 3-19: Optimal number of wavelengths comparison versus cost parameter ratio C_w/C_d of different estimators. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10, \mu = 6$. $C_w = C_d = 100$, $w_0 = 1$, $p_m = 10\%$. [59]

3.5 Chapter 3 Summary

In this chapter, we address the design of a fast-response algorithm for wavelength reconfiguration. Two Bayesian estimators and a stopping-trial sequential estimator proposed in [12] are elaborated to detect changes of traffic arrival statistics. Based on the network transient behaviors of the network, we have shown that the stopping-trial estimator has the shortest detection time for traffic rate changes, and it requires no knowledge of *a priori* probabilities. With continuous assessment, the system reconfigures only when it is necessary. Allowing for the possibility of the addition

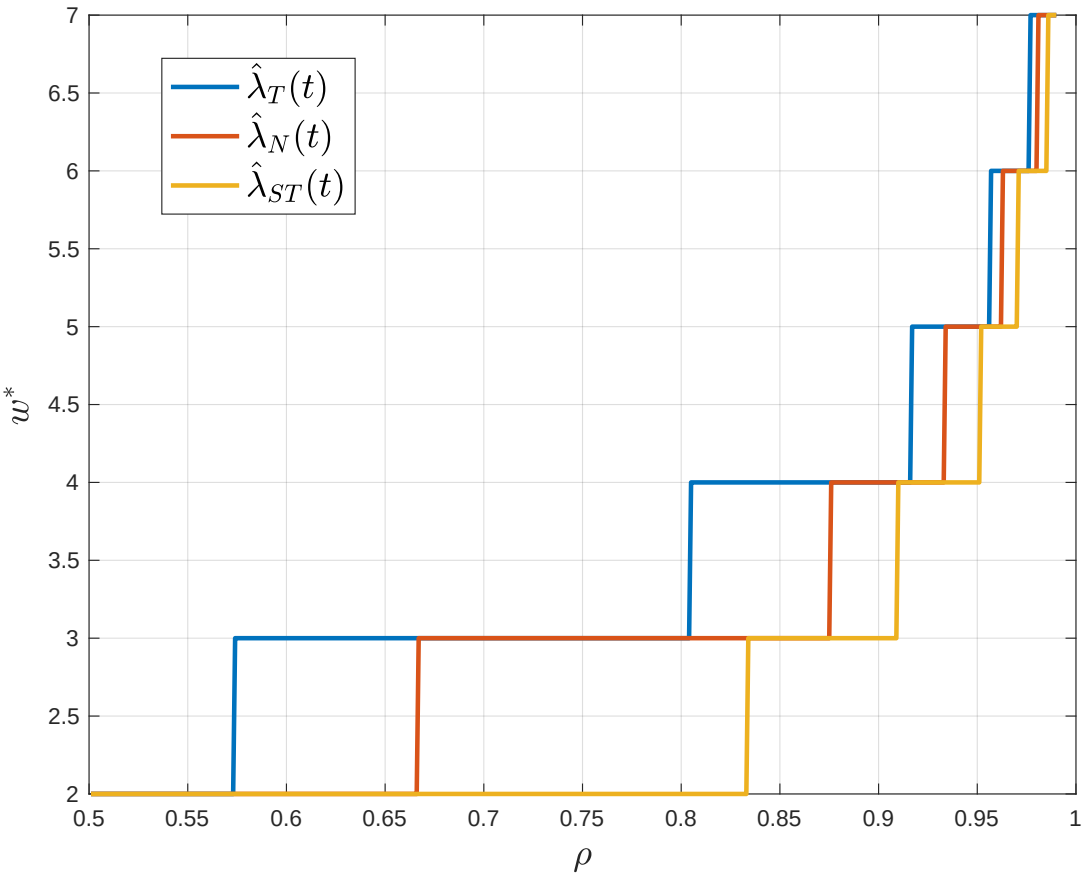

Figure 3-20: Optimal number of wavelengths comparison versus load ρ of different estimators. $\pi_{\lambda_0} = \pi_{\lambda_1} = 0.5, \lambda_0 = 5, \lambda_1 = 10, \mu = 6$. $C_w = C_d = 100$, $w_0 = 1$, $p_m = 10\%$. [59]

and subtraction of multiple wavelengths, the stopping-trial estimator (among all three estimators) requires the smallest number of wavelengths to be reconfigured due to its fastest response that helps to avoid a high peak queueing delay. Our design can reduce queueing delays during traffic surges without over-provisioning, which will reduce network capital expenditures and waste network resources due to erroneous decisions when surges occur.

Chapter 4

Moderate Coherence Time Traffic Environment

The network coherence can vary a lot across different network environments, and a comprehensive analysis of it is a must for the design of cognitive management and control of optical networks. In Chapter 3, a stopping-trial estimator to fast detect traffic changes was examined. When the network coherence time is long, it was shown that the stopping-trial estimator $\hat{\lambda}_{ST}(t)$ has the shortest detection time given the session arrival statistics provide enough confidence for reconfiguration. In this chapter, we keep exploring the efficacy of the stopping-trial estimator $\hat{\lambda}_{ST}(t)$ in the moderate coherence time environment. If the stopping-trial estimator $\hat{\lambda}_{ST}(t)$ works well in the moderate network regime, we can recommend the same wavelength addition and subtraction scheme as the subsequent reconfiguration to deal with traffic surges/drops.

In this chapter, the fast detection in the moderate coherence time regime with a general multi-state Markov traffic model is discussed. We can show the stopping-trial estimator $\hat{\lambda}_{ST}(t)$ can still make a decision in the shortest possible time when the session arrival statistics provide enough confidence for reconfiguration without a predetermined observation time or count. As long as the inter-arrival times of traffic sessions are independent, the stopping-trial estimator can react effectively to the traffic rate changes.

4.1 Validation of Stopping-Trial Estimator

The major difference between the long network coherence time environment and the moderate environment is the frequency of the network traffic changes. As analyzed in Chapter 2, in the long coherence time environment, network traffic changes very slowly so adaption can be done while the traffic is in the same state. The inter-arrival times T_is after the traffic change are a set of I.I.D. variables, and the I.I.D. property can facilitate detection and estimation of the traffic change. However, in the moderate coherence environment, network traffic changes at a moderate rate commensurate with the fastest adaption times. T_is are no longer identically distributed, which raises the question whether the threshold triggering of the random walk is still valid. Therefore, we first need to validate the performance of the stopping-trial estimator $\hat{\lambda}_{ST}(t)$.

In the moderate network coherence time environment, we still observe the inter-arrival times T_i of the traffic arrival process as a sequential test to trigger network reconfigurations for the stopping-trial estimator $\hat{\lambda}_{ST}(t)$. The change in the number of wavelengths will be triggered by the detection of the arrival rate change. Similarly, the random walk in the moderate coherence time environment is the same as the one in the long coherence time but with a set of independent but nonidentically distributed inter-arrival times T_is. Therefore, the random walk is the sum of the each inter-arrival time minus an offset determined by the starting state as in [60]

$$S_J = \sum_{i=1}^{J} (T_i - \frac{1}{\lambda_j}), \tag{4.1}$$

where the process is assumed to start from state λ_j without loss of generality. J is the time that a threshold is crossed.

Notice the above formula for S_J is the same as the random walk in moderate network coherence time but with a set of independent but nonidentically distributed inter-arrival times. Denote the threshold for adding a new wavelength as η_+ and the threshold for tearing down an existing wavelength as η_-. Both η_+ and η_- are

determined by the desired error probabilities. Once S_J crosses a threshold, the corresponding reconfiguration will be made. S_J will then reset, and the offset will be changed from the new state. The wavelength reconfiguration algorithm for the multi-state Markov arrival rate change is shown in Algorithm 2.

Algorithm 2 Stopping-trial wavelength reconfiguration for multi-state Markov changes

Input: arrivals

Output: $m(t)$

if The $(i+1)^{th}$ arrival detected at t **then**
  if $\lambda(t-1) = \lambda_j$ **then**
    $S_n(t) \leftarrow S_n(t-1) + T_i - \frac{1}{\lambda_j}$
    if $S_n(t) < \eta_+$ **then**
      $S_n(t) \leftarrow 0$
      $m(t) \leftarrow m(t-1) + 1$
    else if $S_n(t) > \eta_-$ **then**
      $S_n(t) \leftarrow 0$
      $m(t) \leftarrow m(t-1) - 1$
    else
      $m(t) \leftarrow m(t-1)$
    end if
  end if
else
  $S_n(t) \leftarrow S_n(t-1)$
  $m(t) \leftarrow m(t-1)$
end if

4.1.1 Uncertain Start Counting Point

Our previous work assumes that the random walk counting starts from the point of the traffic change. Therefore, all T_is are identical, and S_n is a random walk.

However, more realistically the counting start point can start at any time, since we cannot predict when traffic changes will occur. Also, the T_is may not be identically distributed since there could be a mix of different λ_is. Fortunately, we can prove that the threshold trigger process with an earlier counting start point is the same as counting starting from the point of traffic change.

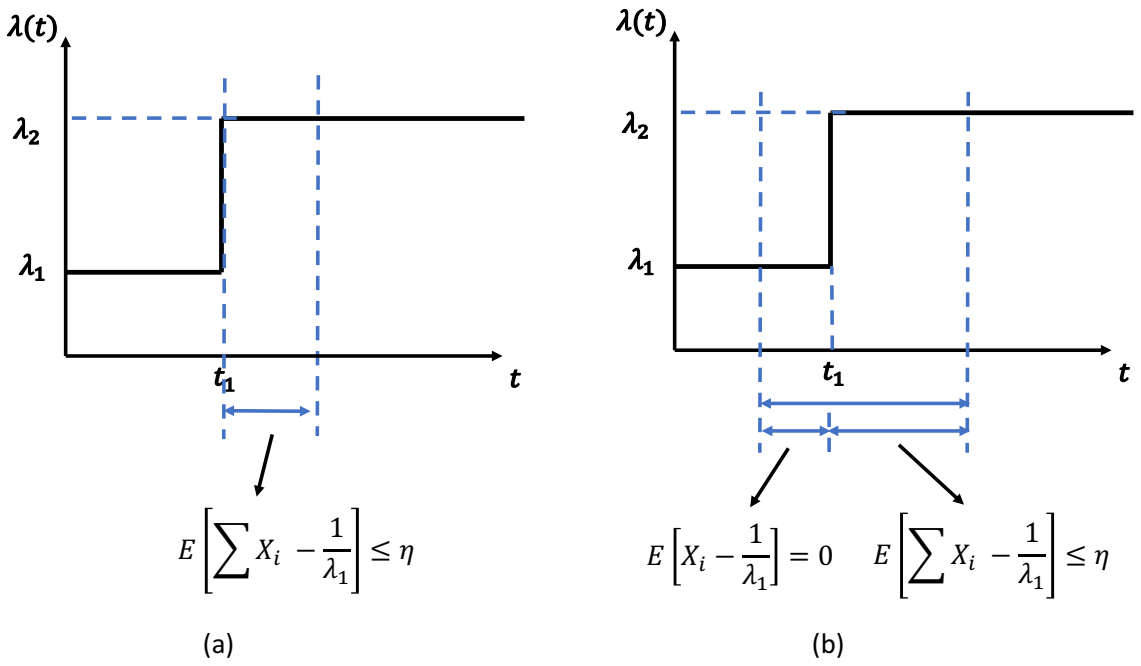

Figure 4-1: The threshold trigger processes with different counting start points. [60]

Without loss of generality, we demonstrate this in a two-state Markov process as shown in Fig. 4-1. When the traffic rate is at λ_1, the average step size is zero as

$$E[T_i - \frac{1}{\lambda_1}] = 0, \tag{4.2}$$

which contributes nothing to the S_n for the threshold crossing on average. Therefore, only the inter-arrival time at rate λ_2 contributes to the threshold crossing, since the average step size is non-zero as

$$E[T_i - \frac{1}{\lambda_1}] = \frac{1}{\lambda_2} - \frac{1}{\lambda_1} < 0. \tag{4.3}$$

The statement also holds if later a traffic drop happens, since the inter-arrival time at λ_1 contributes nothing to the S_n on average. One tricky part is the inter-arrival where the surge occurs. To deal with it, we can first look forward, starting from the epoch of the surge to the next arrival with λ_2. Due to the memoryless property of the exponential distribution, this interval follows an exponential distribution with λ_2. If we look backwards, starting from the epoch of the surge to the previous arrival with λ_2, this interval also follows an exponential distribution with λ_1 due to the reversibility of the Poisson process. It has no impact on S_n on average. Therefore, the threshold trigger process with an earlier counting start point on average is the same as the case of counting starting from the point of traffic change.

4.1.2 Detection Performance in Different Environments

Figure 4-2 shows $\hat{\lambda}_{ST}(t)$ can provide desired wavelength assignment schemes in different network traffic environments. When the coherence time is long or moderate as shown in Fig. 4-2 (a) (b), $\hat{\lambda}_{ST}(t)$ can catch the traffic changes in a possible short time interval before the underlying traffic statistics changes, and the reconfiguration schemes match the requirements of the underlying traffic arrival patterns.

Though $\hat{\lambda}_{ST}(t)$ cannot successfully track the traffic changes within a super-short coherence time as shown in Fig. 4-2 (c), it can still track the trend of the traffic provided the estimation time is shorter than the trend. $\hat{\lambda}_{ST}(t)$ provides fewer frequent reconfigurations since each of the super-fast traffic changes provides, at most, only one observable arrival. Only the overall trend affects reconfigurations, which is the desirable attribute of the algorithm. Coincidentally, this also reduces the network control efforts, since the random walk accumulation and the memory reset upon the detection of $\hat{\lambda}_{ST}(t)$ help to stabilize the reconfigurations, avoiding highly frequent

changes. Even if a false alarm occurs, the error can be quickly corrected.

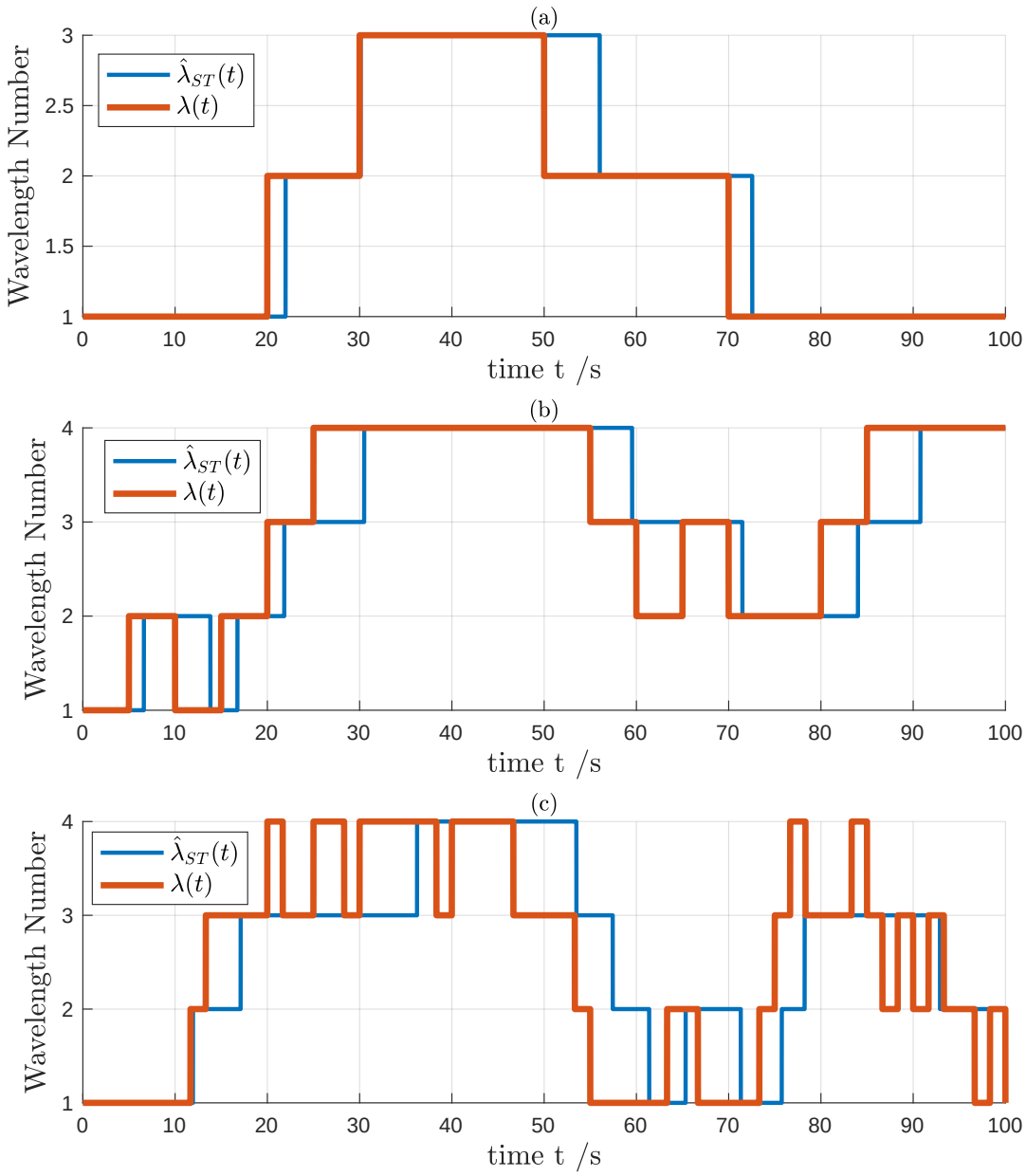

Figure 4-2: The comparison of simulated wavelength assignment of $\hat{\lambda}_{ST}(t)$ and the desired wavelength assignment in different network traffic environments: (a) long network coherence time; (b) moderate network coherence time; (c) short network coherence time. [60]

4.2 Detection Performance in Moderate Network Coherence Time Environment

When the change frequency of the traffic rate is moderate, the network coherence time is similar to the average detection time of $\hat{\lambda}_{ST}(t)$. In this case, the random walk S_J can consist of independent but nonidentically distributed T_is as $\lambda(t)$ is a mixture of different states. Since T_is are no longer identically distributed, we cannot apply Wald's Identity to get the average detection time. Fortunately, we can generalize Wald's Identity with nonidentical but independent random variables as indicated in [4]. [4] provided the Wald's Identity in the form of moment-generating function (MGF), which is hard to use to find the expected detection time. To facilitate the following analysis, we provide the proof in an easily understandable form here as follows:

Statement (Generalized Wald's Identity) For a sequence of nonidentically distributed but independent random variables $z_1, z_2, ..., z_N$, denote

$$Z_N = \sum_{i=1}^{N} z_i, \tag{4.4}$$

where N is a nonnegative integer-valued random variable.

Define $\overline{z}$ as

$$\overline{z} = \frac{1}{N} \sum_{i=1}^{N} E[z_i]. \tag{4.5}$$

We have

$$E[Z_N] = E[N]E[\overline{z}]. \tag{4.6}$$

Proof $\frac{1}{N}$ is a random variable with the distribution of N's inverse distribution,

since N is a nonnegative integer-valued random variable. Then we have

$$\begin{aligned} E[Z_N] &= E[\sum_{i=1}^{N} z_i] && (4.7)\\ &= E[N \cdot \frac{1}{N} \cdot \sum_{i=1}^{N} z_i] && (4.8)\\ &= E[N]E[\frac{1}{N}\sum_{i=1}^{N} z_i] && (4.9)\\ &= E[N]E[\overline{z}], && (4.10) \end{aligned}$$

where Equation (4.9) results from that the expectation of the product of independent random variables equal to the product of individual expectations of random variables.

The generalized Wald's Identity validates the efficacy of $\hat{\lambda}_{ST}(t)$ in the moderate and even short coherence time environments. With the generalized Wald's Identity, we can find the average stopping time $\tau_{1_{ST}}$ of $\hat{\lambda}_{ST}(t)$ in a multi-state Markov process model as

$$\tau_{1_{ST}} = \frac{E[\overline{T_i}]E[S_J]}{E[\overline{T_i - \frac{1}{\lambda_o}}]} \quad (4.11)$$

$$(4.12)$$

where the expectation of the average inter-arrival time $E[\overline{T_i}]$ is

$$E[\overline{T_i}] = E[\frac{1}{J}\sum_{i=1}^{J} E[T_i]] = \frac{1}{J}\sum_{i=1}^{J} E[T_i] \quad (4.13)$$

where the expected

$$E[J] = \frac{E[S_J]}{E[\overline{T_i - \frac{1}{\lambda_o}}]} \quad (4.14)$$

where λ_o is the starting state of the arrival rate. $E[S_J] = \eta_+$ if a traffic surge happens, and $E[S_J] = \eta_-$ if a traffic drop happens.

It is tricky to directly apply the generalized Wald's Identity to get the average detection time as in [59], since J is also a random variable used in both the average inter-arrival time $E[\overline{T_i}]$ and the average step size $E[\overline{T_i - \frac{1}{\lambda_o}}]$. To solve the problem, we can develop the distribution of J from

$$J\overline{T_i} = \sum_{i=1}^{J} E[T_i] \tag{4.15}$$

The analytical solution of J is too complicated and not elegant. Instead, we provide easily calculable analytical upper and lower bounds for $E[J]$ in this work. If we know the range where $\lambda(t)$ potentially moves within as

$$\lambda(t) \in [\lambda_{min}, \lambda_{max}] \tag{4.16}$$

The bounds on $\tau_{1_{ST}}$ for a traffic surge are

$$\lambda_{min} \leq \lambda(t) \leq \lambda_{max} \tag{4.17}$$

$$\Leftrightarrow \frac{1}{\lambda_{max}} \leq E[T_i] \leq \frac{1}{\lambda_{min}} \tag{4.18}$$

$$\Leftrightarrow \frac{\lambda_o \eta_+}{\lambda_o - \lambda_{max}} \leq \tau_{1_{ST}} \leq \frac{\lambda_o \eta_+}{\lambda_o - \lambda_{min}} \tag{4.19}$$

Figure 4-3 shows the potential range of the average detection time τ_1 of $\hat{\lambda}_{ST}(t)$ versus the probability of missed detection for three different arrival rate ranges. When the traffic arrival rate changes into a higher state, the average detection time becomes shorter to make reconfiguration faster, which reflects the adaptive detection time of $\hat{\lambda}_{ST}(t)$. A low missed detection probability requires a long detection time on a traffic surge. It is important to pick a good error probability for the threshold crossing,

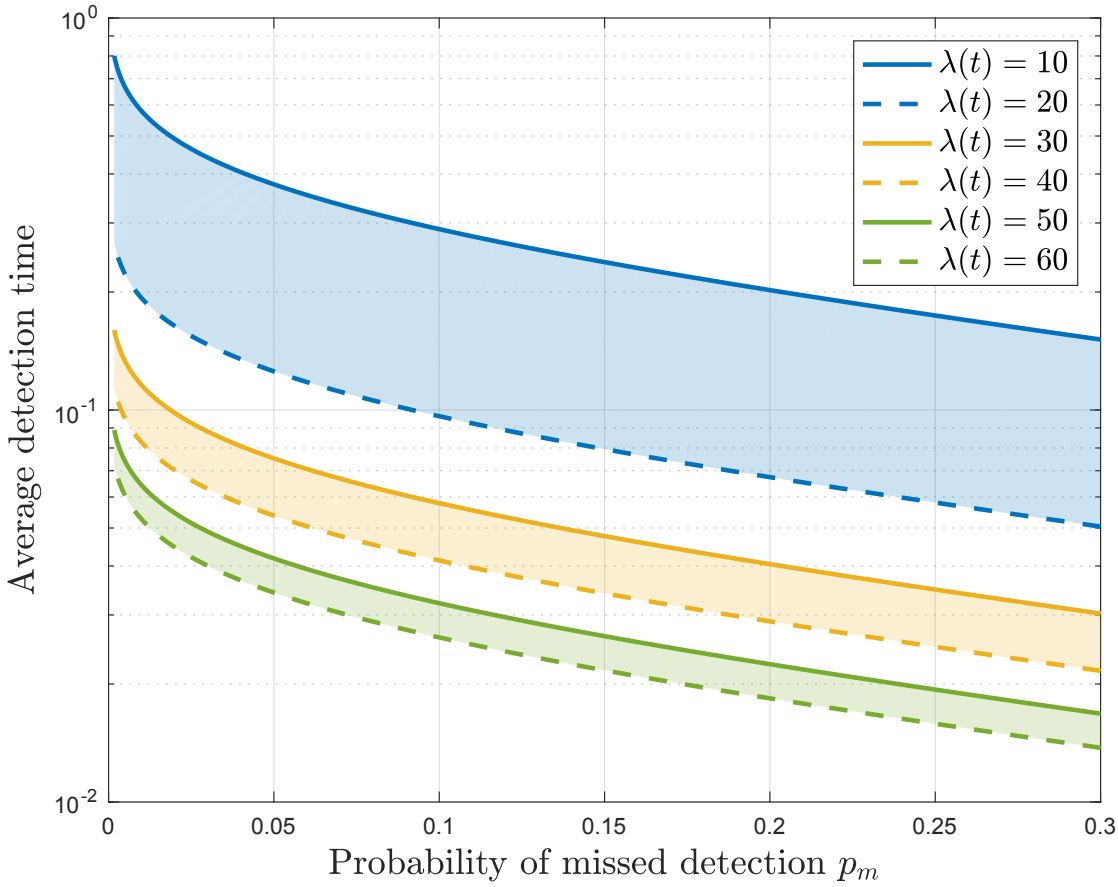

Figure 4-3: The range of the average detection time τ_1 of $\hat{\lambda}_{ST}(t)$ versus probability of missed detection (crossing thresholds) for different arrival rate ranges. [60]

since we want to maintain the desirable short detection time as well as the high reconfiguration accuracy to reduce the burdens of network management and control.

The beauty of the stopping-trial estimator $\hat{\lambda}_{ST}(t)$ is its capability to provide reconfiguration without assuming any detailed traffic model or inference from the network traffic statistics. It only makes a reconfiguration decision when it is necessary to do so at the shortest possible time. Compared to other commonly used estimators requiring the I.I.D. properties of traffic arrivals, $\hat{\lambda}_{ST}(t)$ can respond to the traffic changes swiftly as long as the inter-arrival times of traffic transactions are independent. Also, the adaptable detection time of $\hat{\lambda}_{ST}(t)$ enables the system to reconfigure on a fast

time scale. A higher traffic arrival rate yields a shorter detection time as desired.

4.3 Reconfiguration in Moderate Coherence Time Traffic Environment

Since the stopping-trial estimator $\hat{\lambda}_{ST}(t)$ works well in the moderate network regime, we recommend the same wavelength addition and subtraction scheme after the change detection by the stopping-trial estimator $\hat{\lambda}_{ST}(t)$. As shown in Chapter 2, the stopping-trial estimator $\hat{\lambda}_{ST}(t)$ provides the short response time to the traffic changes, and it can save costs by reducing the queueing delays. With continuous assessment enabled, the system reconfigures only when it is necessary. Further configurations will be discussed in Chapter 6.

4.4 Chapter 4 Summary

In this chapter, we validate the performance of the stopping-trial estimator in the moderate network coherence time environment. The stopping-trial estimator can provide the desired wavelength assignment scheme and make reconfiguration decisions at the shortest possible time with independent but not necessarily identically distributed inter-arrival times of traffic sessions.

Chapter 5

Short Coherence Time Traffic Environment

In the short coherence time traffic environment, network traffic can change very rapidly. It is hard to detect every single traffic change, since the detection time of the estimator can be much longer than the network coherence time. In Chapter 3 and Chapter 4 respectively, we have shown a stopping-trial estimator $\hat{\lambda}_{ST}(t)$ to fast detect traffic changes work well in both the long network coherence time environment and the moderate coherence time environment. Wavelength addition/subtraction with continuous assessments enabled is recommended as the subsequent reconfiguration to deal with the traffic surges/drops. However, $\hat{\lambda}_{ST}(t)$ cannot track well the traffic changes within a super-short coherence time. If the coherence time is extremely short, the detection may no longer converge. In this case, we can only try to predict the trend when the session arrival statistics provide enough confidence that the fast traffic changes exhibit an underlying trend. So, the adaptions are limited to trends rather than to detail changes.

In this chapter, we discuss the traffic trend detection in the short coherence time traffic environment [60]. In the short coherence time regime, the coherence time can change gently or abruptly depending on the shift in the offered traffic's statistics. A transient condition prevails until the system arrives at a new steady state. We model the transient behavior of such network traffic drifts towards convergence to

a new steady state analytically and validate the feasibility of the traffic prediction. For a special case of fast-changing traffic where the traffic rate increases/decreases monotonically in a linear model, we develop a sequential maximum likelihood estimator that can predict the traffic trend in the super-short network coherence time environment. It can sufficiently estimate the traffic trend with a reasonable number of arrivals and trigger reconfigurations.

5.1 Fast Detection in Short Network Coherence Time Environment

When the changing frequency of the traffic rate is high, the network coherence time is much shorter than the detection time of the estimator. The simulation results of the previous chapter have shown that it is hard to track all the super-fast changes. Even though an optimal minimum mean square estimator on $\lambda(t)$ is provided in [17], the detection of the fast-changing traffic is still challenging if the details of $\lambda(t)$ is unknown. The optimal minimum mean square estimator requires the detailed model of $\lambda(t)$, which is hard to find with only several arrivals. As the minimum mean square estimator in [17] is too complicated, a simple way to approximate the real-time arrival rate is to use the average arrival rate over the time as

$$\lambda = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} \lambda(t) dt, \tag{5.1}$$

where t_1 is the start time and t_2 is the end time.

With our multi-state traffic arrival model, we define Λ_i as the average arrival rate over the corresponding inter-arrival time T_i, and we have

$$\Lambda_i = \frac{1}{X_i} \int_{X_i} \lambda(t) dt. \tag{5.2}$$

For n inter-arrival times $T_1, T_2, ..., T_n$, we assume the corresponding arrival rates are $\Lambda_1, \Lambda_2, ..., \Lambda_n$. This is because the real-time arrival rate changes too fast to be reflected in an inter-arrival time interval so that a quasi-statically constant intensity substitute Λ_i is a good approximation. When the observation interval is the sum of the several inter-arrival times, we have

$$S_n = \sum_{i=1}^{n}(T_i - \frac{1}{\Lambda_o}) \tag{5.3}$$

$$\Leftrightarrow \sum_{i=1}^{n} T_i = S_n + \frac{n}{\Lambda_o}, \tag{5.4}$$

where Λ_o is the starting arrival rate before the threshold is crossed. The distribution of S_n follows the convolution of n exponential distributions where each one has a unique rate, as shown in Equation (5.3). Therefore, the distribution of S_n is

$$f_{S_n}(t) = \sum_{i=1}^{n} \frac{\Lambda_1 ... \Lambda_n}{\prod_{j=1, j\neq i}^{n}(\Lambda_j - \Lambda_i)} e^{-\Lambda_i t}, t > 0. \tag{5.5}$$

It is extremely difficult to track the totally random process of the arrival rate transition in the short network coherence time regime, since the network arrival rate changes again almost from one arrival to the next. Though an optimal minimum mean square estimator on $\lambda(t)$ is provided in [17], its performance on a fast-changing total random process can be bad. It is also not necessary because the result is the short-term average of these rates that will affect the length of the queues. However, if the fast-changing traffic follows a certain pattern, we can try to predict the trend, which is the relevant parameter to track and affects the queues in the network.

A simple nontrivial case is that the traffic changes so fast that the traffic arrival rate increases/decreases monotonically as a linear function. Intuitively, the traffic is drifting in the direction of the embedded Markov chain, and we define the duration of the traffic's drift in one direction as the network drifting time. The network usually drifts in a direction when the embedded Markov chain is converging to a new steady

state as the result of transition rate changes. Though the changing frequency of the steady state is low practically, it can cause severe network congestion and cause burdens for subsequent reconfigurations if it is not detected promptly. Therefore, it is necessary to analyze the transient behaviors of network drifts. Moreover, we should try to predict the traffic trend and develop the corresponding reconfiguration algorithm for network traffic drifts. Sometimes, a network traffic drifting can happen as the normal transition of the Markov process without the change of the steady state. However, such drifts are temporary and will not last for a long time, which will not be discussed in this work. Other traffic rate changing patterns can be stuided

5.2 Network Drifting Time

When the network traffic drifts due to a shift of the Markov process transition rates, we want to know the transient behavior of the system on how the drift converges and how long it takes to achieve the new steady state. Both metrics are determined by the new transition probability matrix P. The transition probability matrix can be represented with each entry p_{ij} indicating the transition probability from state λ_i to state λ_j. It has the same form as the in the stochastic matrix P_{sample} of $M/M/m(t)$ queue in Chapter 3. Denote b wavelengths that are assigned during the drift. If the value does not vary, then the new transition probability matrix $P =$

$$\begin{bmatrix}
1-\lambda\delta & \lambda\delta & 0 & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\mu\delta & 1-(\lambda+\mu)\delta & \lambda\delta & 0 & \dots & \dots & \dots & \dots & \dots & \dots \\
0 & 2\mu\delta & 1-(\lambda+2\mu)\delta & \lambda\delta & 0 & \dots & \dots & \vdots & \dots & \dots \\
\vdots & \ddots & \ddots & \dots & \vdots & \vdots & \vdots & \vdots & \vdots & \dots \\
\dots & \dots & \dots & 0 & b\mu\delta & 1-(\lambda+b\mu)\delta & \lambda\delta & 0 & \dots & \dots \\
\dots & \dots & \dots & \dots & 0 & b\mu\delta & 1-(\lambda+b\mu)\delta & \lambda\delta & 0 & \dots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & 0 & b\mu\delta & 1-b\mu\delta
\end{bmatrix}.$$

The convergent rate to achieve a new steady state is largely determined by the

second largest eigenvalue in magnitude of P, since the largest eigenvalue is always 1 and corresponds to the steady state [21]. A larger second largest eigenvalue of P in magnitude yields a lower convergent rate, which means the system takes a longer time to converge. Apart from the convergent rate, the total convergent time also greatly depends on the initial steady-state probability distribution. If the final steady-state probability distribution differs more from the initial distribution, it will take a longer time to converge.

We can find the settling time τ_2 to the new steady state with the eigenvalues of P, the eigenvectors of P, and the initial steady state probability distribution. We assume P is invertible to facilitate the analysis, and P has l distinct eigenvalues. To get the eigenvalues and the eigenvectors of P, we decompose P as

$$P = U\Gamma U^{-1} \tag{5.6}$$

where Γ is the diagonal matrix with entries $\gamma_1, ..., \gamma_l$ as all the eigenvalues as

$$\Gamma = \begin{bmatrix} \gamma_1 & 0 & 0 & \dots & 0 \\ 0 & \gamma_2 & 0 & \dots & 0 \\ 0 & 0 & \gamma_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \gamma_l \end{bmatrix}.$$

U is a matrix whose columns are the corresponding eigenvectors of the eigenvalues on the diagonal of Γ. U^{-1} is the inverse of U.

The transition probability matrix P must have the largest eigenvalue with the value as 1 [21]. It is also the only value equals to 1, since all l eigenvalues are assumed distinct. Without loss of generality, denote $\gamma_1 = 1$, then $|\gamma_i| < 1$ for $i \neq 1$, and denote γ_2 as the second largest eigenvalue in magnitude.

Denote the initial steady-state probability distribution $\Pi(0)$ when the network drift begins as

$$\Pi(0) = [\pi_1(0), \pi_2(0), ..., \pi_l(0)]. \tag{5.7}$$

The new steady state is reached when time tends to infinity. Assume P does not change again after the drift happens. Denote the new steady-state distribution $\Pi(\infty)$ as

$$\Pi(\infty) = [\pi_1(\infty), \pi_2(\infty), ..., \pi_l(\infty)]. \tag{5.8}$$

With an arbitrarily small positive quantity ϵ and a sample unit time δ, we now can represent $\Pi(0)$ approximately close to $\Pi(\infty)$ within a time length of τ_2 as

$$\begin{aligned}
||\Pi(0)P^{\frac{\tau_2}{\delta}} - \Pi(\infty)|| &\leq \epsilon & (5.9)\\
\Leftrightarrow ||\Pi(0)(U\Gamma U^{-1})^{\frac{\tau_2}{\delta}} - \Pi(0)(U\Gamma U^{-1})^{\infty}|| &\leq \epsilon & (5.10)\\
\Leftrightarrow ||\Pi(0)U\Gamma^{\frac{\tau_2}{\delta}}U^{-1} - \Pi(0)U\Gamma^{\infty}U^{-1}|| &\leq \epsilon & (5.11)\\
\Leftrightarrow ||\Pi(0)U(\Gamma^{\frac{\tau_2}{\delta}} - \Gamma^{\infty})U^{-1}|| &\leq \epsilon & (5.12)
\end{aligned}$$

where

$$\Gamma^{\frac{\tau_2}{\delta}} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & \gamma_2^{\frac{\tau_2}{\delta}} & 0 & \dots & 0 \\ 0 & 0 & \gamma_3^{\frac{\tau_2}{\delta}} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \gamma_l^{\frac{\tau_2}{\delta}} \end{bmatrix}, \Gamma^{\infty} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & & \dots & 0 \end{bmatrix}.$$

By solving the inequality (5.12), we can get the settling time τ_2. We can also get an elegant analytical upper bound on τ_2 with the second largest eigenvalue γ_2 by applying an upper bound on

$$\Gamma^{\frac{\tau_2}{\delta}} - \Gamma^{\infty} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & \gamma_2^{\frac{\tau_2}{\delta}} & 0 & \dots & 0 \\ 0 & 0 & \gamma_3^{\frac{\tau_2}{\delta}} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \gamma_l^{\frac{\tau_2}{\delta}} \end{bmatrix} \leq \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & \gamma_2^{\frac{\tau_2}{\delta}} & 0 & \dots & 0 \\ 0 & 0 & \gamma_2^{\frac{\tau_2}{\delta}} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \gamma_2^{\frac{\tau_2}{\delta}} \end{bmatrix}. \tag{5.13}$$

To facilitate the notation, we denote a matrix W as

$$W = \Pi(0)U = [w_1, w_2, ..., w_l]. \tag{5.14}$$

We can get an upper bound on the left-hand side of equation (5.12) as

$$\begin{aligned} \Pi(0)U(\Gamma^{\frac{\tau_2}{\delta}} - \Gamma^{\infty})U^{-1} &\leq [0, w_2\gamma_2^{\frac{\tau_2}{\delta}}, ..., w_l\gamma_2^{\frac{\tau_2}{\delta}}]U^{-1} && (5.15) \\ &= \gamma_2^{\frac{\tau_2}{\delta}}[0, w_2, ..., w_l]U^{-1} && (5.16) \\ &\triangleq \gamma_2^{\frac{\tau_2}{\delta}} W'U^{-1}. && (5.17) \end{aligned}$$

where we denote $W' = [0, w_2, ..., w_l]$ to facilitate the notation. Notice the only difference between W' and W is that the first value in W' is 0, but all other values are the same as that in W in the corresponding positions.

Then we have

$$\begin{aligned} ||\Pi(0)U(\Gamma^{\frac{\tau_2}{\delta}} - \Gamma^{\infty})U^{-1}|| &\leq ||\gamma_2^{\frac{\tau_2}{\delta}} W'U^{-1}|| && (5.18) \\ &\leq ||\gamma_2^{\frac{\tau_2}{\delta}} W'|| \cdot ||U^{-1}|| && (5.19) \\ &= \gamma_2^{\frac{\tau_2}{\delta}} ||W'|| \cdot ||U^{-1}||, && (5.20) \end{aligned}$$

where $||W'||$ is the Euclidean norm (or 2-norm) of the vector W', and $||U^{-1}||$ is the 2-norm of the matrix U^{-1}. Using the upper bound in (5.20), an upper bound on τ_2 is

$$\begin{aligned}
\gamma_2^{\frac{\tau_2}{\delta}} ||W'|| \cdot ||U^{-1}|| &\leq \epsilon && (5.21)\\
\Leftrightarrow s_2^{\frac{\tau_2}{\delta}} &\leq \frac{\epsilon}{||W'|| \cdot ||U^{-1}||} && (5.22)\\
\Leftrightarrow \frac{\tau_2}{\delta} \log s_2 &\leq \log \frac{\epsilon}{||W'|| \cdot ||U^{-1}||} && (5.23)\\
\Leftrightarrow \tau_2 &\leq \log_{\gamma_2} \frac{\epsilon\delta}{||W'|| \cdot ||U^{-1}||} && (5.24)\\
\Leftrightarrow \tau_2 &\leq \log_{\gamma_2} \frac{\epsilon\delta}{||W'|| \sigma_{max}(U^{-1})}, && (5.25)
\end{aligned}$$

where $\sigma_{max}(U^{-1})$ represents the largest singular value of U^{-1}.

To evaluate the convergent times of different initial steady states, we use the idea of Kullback-Leibler divergence [18] to represent the distance between the initial steady-state probability distribution and the new steady-state probability distribution after the network drift $Dist.(\Pi(0)||\Pi(\infty))$ as

$$Dist.(\Pi(0)||\Pi(\infty)) = \sum_{x \in \mathcal{X}} \Pi(0) \log \left(\frac{\Pi(0)}{\Pi(\infty)} \right), \tag{5.26}$$

where both $\Pi(0)$ and $\Pi(\infty)$ are defined on the same probability space $\mathcal{X}$. The distance $Dist.(\Pi(0)||\Pi(\infty))$ is a measure of how $\Pi(0)$ is different from $\Pi(\infty)$, where we set $\Pi(\infty)$ as the reference probability distribution. A larger $Dist.(\Pi(0)||\Pi(\infty))$ represents a larger difference.

Figure 5-1 shows that our analytical upper bound on the settling time τ_2 can approximate τ_2 well. The settling time τ_2 increases with the increase of the distance between the initial steady-state probability distribution and the new steady-state probability distribution after the network traffic drift. Besides, a larger second largest eigenvalue in magnitude γ_2 of the probability transition matrix results in a longer settling time as expected. A small amount of increase in γ_2 can result in a noticeable

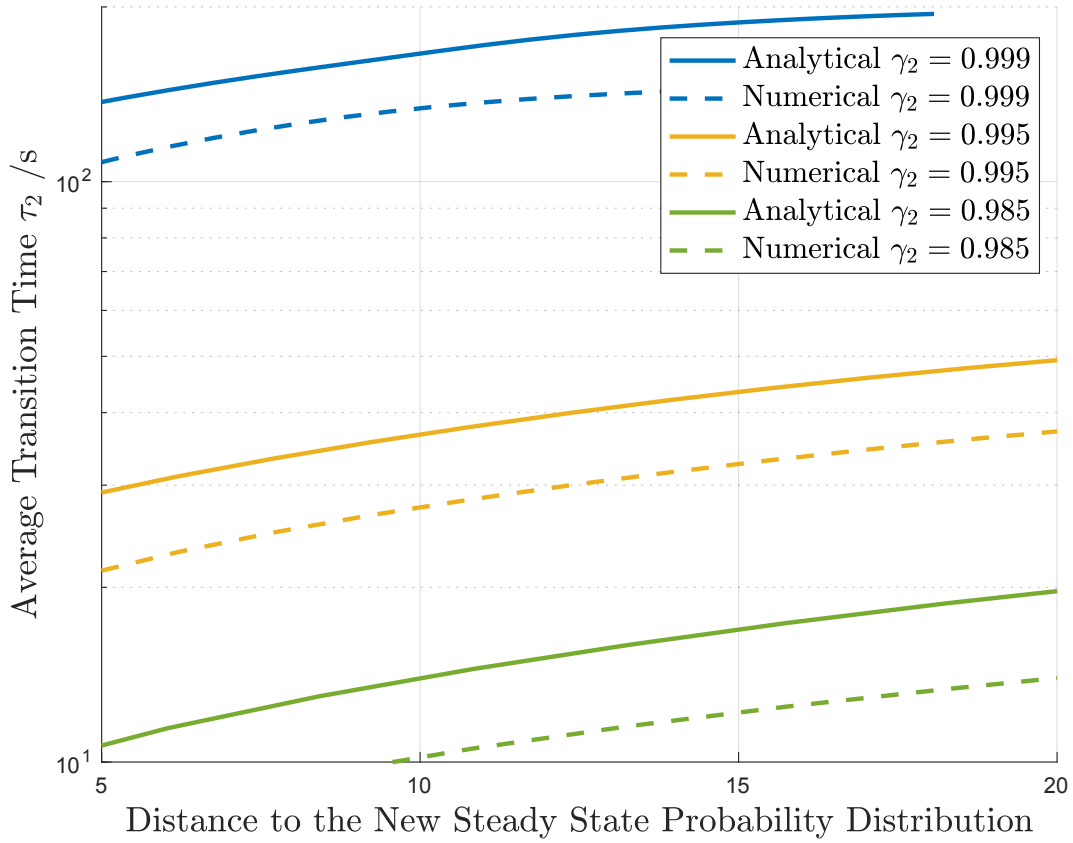

Figure 5-1: Analytical upper bounds and numerical results on the settling time τ_2 versus different distances between the initial steady-state probability distribution and the new steady-state probability distribution after the network drift with different convergent rates (the second largest eigenvalue in magnitude of the probability transition matrix). $l = 100, a_{i,i+1} = 10\ (0 < i < l-1), \delta = 0.01s, \epsilon = 10^{-5}$. [60]

increase in the settling time due to the iterative multiplication of the transition probability matrix. By modeling the transient behavior of the network drifts analytically, we can know whether the network traffic drift will maintain a long enough network settling time for us to predict the traffic trend. It will be good if we can predict the trend before the new steady state is reached and allocate resource in advance to avoid the foreseeable network congestion.

5.3 Prediction of the Network Traffic Drifting Trend

We depict a simple but nontrivial case of the network traffic drifts in a linear model, where the network coherence time is so short that the arrival rate changes at every epoch with the same rate. Notice here the linear model is a special case to demonstrate the possibility of estimating the trend of the traffic. Other changing models, such as the exponential model or even more complicated models, can be discussed as the extension of this work. Without loss of generality, we only discuss the process of increasing network traffic drifts in the following sections since the decreasing process is the same, just in reverse. We assume the arrival rate increases in a constant slope of k as

$$\lambda(t) = \Lambda_o + kt,\ k > 0, \tag{5.27}$$

where $k > 0$ to indicate it is an increasing process, and Λ_o is the starting arrival rate.

Our goal is to determine k with a given time interval of arrivals. We can estimate k by the maximum likelihood estimation based on the probability of the arrivals sequentially. To reduce the computation complexity, we limit k to a certain range as

$$k_{min} \leq k \leq k_{max}. \tag{5.28}$$

If k is too large such as tending to infinity, the traffic arrival rate increases too sharply to be reconfigured. If k is too small such as tending to zero, there are almost no traffic arrival change and no reconfiguration is needed. The probability density function of an inhomogeneous Poisson process with arrivals at epoch $t_1, t_2, ..., t_n$ in a time interval with length of T_0 as described in [21] is

$$f(t_1, t_2, ..., t_n|\lambda(t)) = \Big[\prod_{i=1}^{n} \lambda(t_i)\Big] \exp\Big\{ - \int_0^{T_0} \lambda(\eta) d\eta \Big\}. \tag{5.29}$$

The probability that n arrivals happen in the time interval $(0, T_0)$ is

$$Pr(N = n) = \int_{0<t_1<...<t_n<T_0} f(t_1, ..., t_n|\lambda(t))dt_1...dt_n, \tag{5.30}$$

where $0 < t_1 < ... < t_n < T_0$ indicate the order of arrivals, which cannot be ignored.

Therefore, we have the maximum likelihood estimator as

$$\hat{k} = \text{argmax}_{k_{min}\leq k\leq k_{max}} \quad f(t_1, t_2, ..., t_n|\lambda(t)). \tag{5.31}$$

Given the linear arrival rate change model as $\lambda(t) = \Lambda_o + kt$, we get $\hat{k}$ as

$$\frac{d}{dk} f(t_1, t_2, ..., t_n|\lambda(t)) = 0 \tag{5.32}$$

$$\Leftrightarrow \frac{d}{dk} f(t_1, t_2, ..., t_n|\Lambda_o + kt) = 0 \tag{5.33}$$

$$\Leftrightarrow \frac{d}{dk} \prod_{i=1}^{n} (\Lambda_o + kt_i) \exp\Big\{ - \int_0^{T_0} \lambda(\eta)d\eta \Big\} = 0. \tag{5.34}$$

Obviously, the $\exp\{\cdot\}$ part is the area of density function during $[0, T_0]$. By taking log of the equation (5.34), we have

$$\frac{d}{dk} \Big[\sum_{i=1}^{n} \ln(\Lambda_o + kt_i) - \int_0^{T_0} \lambda(\eta)d\eta \Big\} = 0 \tag{5.35}$$

$$\Leftrightarrow \frac{d}{dk} \Big[\sum_{i=1}^{n} \ln(\Lambda_o + kt_i) - (\Lambda_o + \frac{kT_0}{2})T_0 \Big] = 0 \tag{5.36}$$

$$\Leftrightarrow \Big[\sum_{i=1}^{n} \frac{t_i}{\Lambda_o + kt_i} \Big] - \frac{T_0^2}{2} = 0 \tag{5.37}$$

It is hard to find the analytical results of k from equation (5.37), and we can also prove that there is no closed-form solution for the roots of a fifth or higher degree polynomial equation by the Abel-Ruffini Theorem [47]. Instead, we can use

the numerical methods to find the only positive root.

Apart from finding the only possible root with numerical methods, we can find closed-form analytical bounds on $\hat{k}$ to facilitate the calculation. For the lower bound, we apply Arithmetic Mean-Harmonic Mean (AM-HM) inequality as

$$\begin{aligned} \sum_{i=1}^{n} \frac{t_i}{\Lambda_o + kt_i} &= \sum_{i=1}^{n} \frac{1}{\frac{\Lambda_o}{t_i} + k} && (5.38) \\ &\geq \frac{n^2}{\sum_{i=1}^{n} (\frac{\Lambda_o}{t_i} + k)} && (5.39) \end{aligned}$$

Combining with the equation in (5.37), we have

$$\begin{aligned} \sum_{i=1}^{n} (\frac{\Lambda_o}{t_i} + k) &\geq \frac{2n^2}{T_0^2} && (5.40) \\ \Leftrightarrow k &\geq \frac{\frac{2n^2}{T_0^2} - \sum_{i=1}^{n} \frac{\Lambda_o}{t_i}}{n} && (5.41) \\ \Leftrightarrow k &\geq \frac{2n}{T_0^2} - \frac{\Lambda_o}{n} \sum_{i=1}^{n} \frac{1}{t_i}. && (5.42) \end{aligned}$$

We can use the proof by contradiction to find an upper bound. Let's first make the assumption on k as $k > \frac{2n}{T_0^2} - \frac{\Lambda_0}{t_n}$. Given the fact $0 < t_1 < t_2 < ... < t_n$, we have

$$\frac{\Lambda_0}{t_1} > \frac{\Lambda_0}{t_2} > ... > \frac{\Lambda_0}{t_n} > 0. \quad (5.43)$$

Therefore, we have

$$k + \frac{\Lambda_0}{t_i} \geq k + \frac{\Lambda_0}{t_n} \tag{5.44}$$

$$> \frac{2n}{T_0^2} \tag{5.45}$$

$$\Leftrightarrow \frac{1}{k + \frac{\Lambda_0}{t_i}} < \frac{T_0^2}{2n} \tag{5.46}$$

$$\Leftrightarrow \sum_{i=1}^{n} \frac{1}{k + \frac{\Lambda_0}{t_i}} < \frac{T_0^2}{2}. \tag{5.47}$$

where the inequality (5.47) contradicts to the equation (5.37). Therefore, our assumption $k > \frac{2n}{T_0^2} - \frac{\Lambda_0}{t_n}$ is wrong, and it thus gives us an upper bound on $\hat{k}$ as

$$k \leq \frac{2n}{T_0^2} - \frac{\Lambda_0}{t_n}. \tag{5.48}$$

It is interesting to find that the forms of the upper bound and the lower bound on $\hat{k}$ are very similar. Both of them are $\frac{2n}{T_0^2}$ minus a value. $\frac{2n}{T_0^2}$ conveys the general information of the length of the observation interval and the counts of arrival in the interval. The value that makes the difference conveys more detailed information on the epochs of the arrivals. Compared to the lower bower, the upper bound only uses the information of the last arrival time.

Figure 5-2 shows the slope estimation of our sequential maximum likelihood estimator $\hat{k}$ in a time period of T_0 with different random numbers of arrivals. It is the average result over 200 runs, where the probability distribution of the arrivals is the same but the arrival samples in each run are randomly generated. The sequential maximum likelihood estimator $\hat{k}$ can estimate the slope sufficiently well with a reasonable number of arrivals, even though there is a bias as shown in Fig. 5-2. The bias comes from the exponential part of the inhomogeneous poisson process. The estimated value is above the true value, which indicates that the estimator estimates the trend aggressively during the fast-rate change. The aggressive estimation can help to allocate more resources in advance to avoid network performance degrada-

tion. However, it should be carefully handled, because the overestimation will incur large over-provisioning, which will be costly.

Both the lower bound $\hat{k}_{lower}$ and the upper bound $\hat{k}_{upper}$ together give a good approximation to $\hat{k}$ analytically. The difference between the upper bound and the lower bound is the product of the value of the starting arrival rate and the difference between the average of the reciprocals of the arrival epochs and the reciprocal of the latest arrival epoch as $\Lambda_o(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{t_i} - \frac{1}{t_n})$. Since the upper bound is simpler and only depends on n and t_n (only the last arrival time is needed), it is a useful exact analytical approximation to the optimum estimate. The convex curve shape indicates that fewer samples make the results inaccurate, and more samples can cause overfitting.

With the prediction of the network traffic drift trend, we can design the proper reconfiguration algorithm and allocate more resources in advance to avoid potential traffic congestion. This enables the system to quickly and accurately adapt to network traffic conditions to save overly jittery control efforts, minimize queueing delays due to late reconfigurations, and use network resources efficiently.

5.4 Summary of Chapter 5

In this chapter, we discuss the traffic detection and predict in the short coherence time environment. We model the transient behavior of the network traffic drifts towards convergence to a new steady state analytically for the super-fast traffic rate changes to validate the feasibility of the traffic prediction. Given a specific network drift, we provide a traffic trend estimation technique for the super-fast traffic rate changes approximated by a simple but nontrivial linear model. The linear model is a special case of the fast-changing rate where the arrival rate changes at every epoch with the same rate. The rate changes can be more complicated as in the exponential model or other uncommon models. With a reasonable number of arrivals, the sequential maximum likelihood estimator can estimate the network traffic drifting trend in a linear model well and enable reconfigurations in advance to minimize congestion.

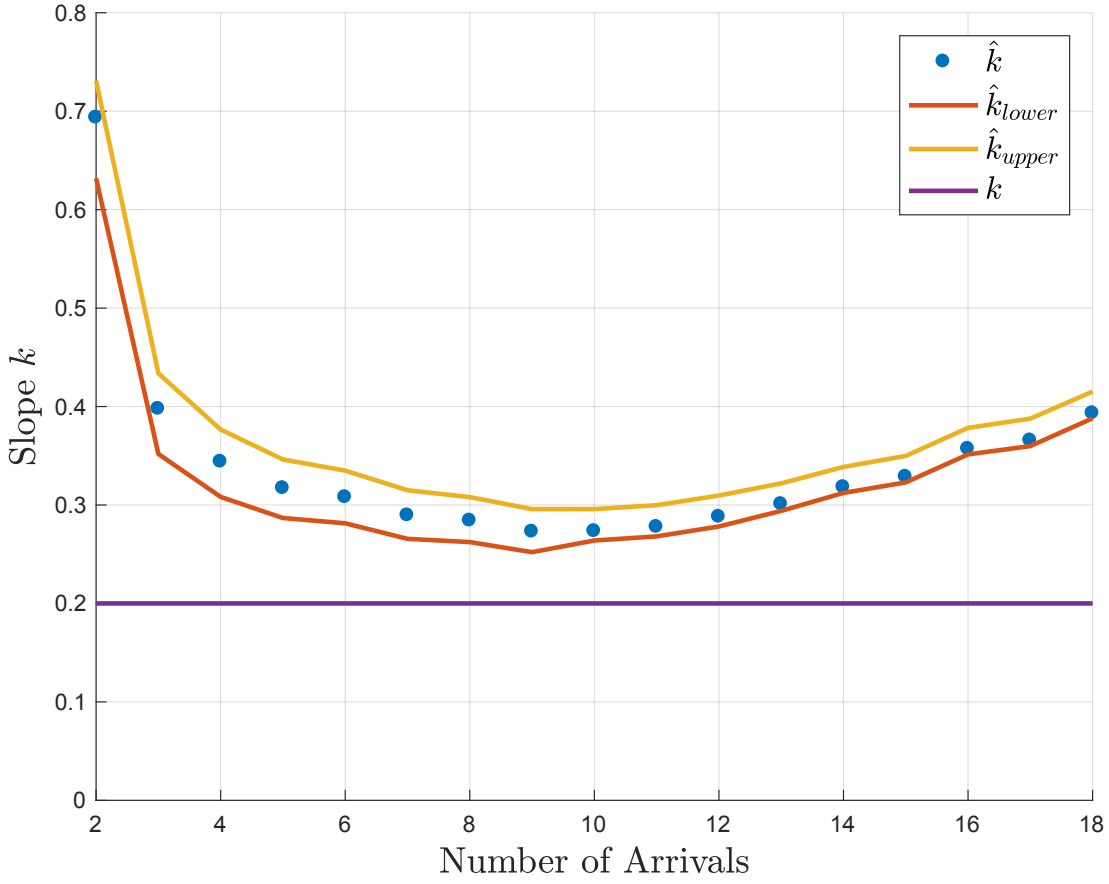

Figure 5-2: Slope estimation of the network traffic drifting trend in a linear increasing model versus different number of arrivals over 200 runs. The probability distribution of the arrivals is the same but each time the arrival samples will be randomly generated. $\Lambda_o = 5, k = 0.2$. [60]

With the traffic trend predicted, network management and control efforts can be more efficient and thus more affordable to meet the requirements of future dynamic traffic.

Chapter 6

Rerouting

When increasing traffic cannot be fully handled by fast wavelength reconfiguration, new reconfigurations should be made to maintain users' quality of service. With the traffic changes detected, three reconfiguration options, in order of preference, are: 1. Lighting up a wavelength in the same primary lightpath between a node pair; 2. Rerouting incremental traffic to a secondary path with open wavelengths between the node pair; 3. Lighting up a new fiber with multiple wavelengths together with optical switching to accommodate overflowing traffic. In the previous chapters, we have proposed a scheme for fast wavelength reconfigurations after the detection and estimation of the dynamic traffic. In this chapter, we further investigate the design of a rerouting algorithm. The time to perform the new fiber setup is also mentioned.

When the incoming traffic has a high queueing delay that exceeds the threshold, the traffic has to go through secondary paths to the destination node, which is called rerouting. Both the primary path of routing and the secondary paths for rerouting are generated by the shortest-path algorithm. Hence, the primary path is the shortest path between the source node and the destination node. To achieve a high successful rate of rerouting, the paths for rerouting are recommended to have a small number of hops and be disjoint with other busy paths. high edge-connectivity network topology is preferred. With the shortest-path algorithm, wavelength reservation for the traffic to be rerouted is not recommended to achieve a good utilization of resources. We adopt the triggering by a threshold on the queueing delay due to its operating sim-

plicity rather than other complicated algorithms, which greatly reduce control efforts. The reason to choose the queueing delay on the primary path is because it is directly measurable by the queue size, which is determined by the current traffic situation and the network configurations. With the same queueing delay threshold, the schedule holder can hold different numbers of transactions for traffic with different sizes when the rerouting is triggered. It is also shown in [29, 30, 26] that partitioning of resources will improve average normalized delay, where the average delay is the performance metric. Per [26], if the network has the mixed traffic, where each traffic category follows an exponential distribution with the corresponding average size, the optimal wavelength assignment to minimize the average delay is to allocate the wavelengths proportioned by the ratio of average traffic sizes. Based on this conclusion in [26] and our analysis of the number of transactions in the schedule holder using the real-time delay deadline as the metric, we make the conjecture that traffic with a great spread of size differences should be split into multi-classes with resources partitioned for rerouting, and the delay for any session should be normalized by its transmission time. Though it may result in different splits as the metrics in this work and in [26] are different, the idea of splitting the traffic to relax the delay requirement will hold in both cases. Therefore, more large transactions can be transmitted on the primary path for better network utilization.

6.1 Rerouted Traffic and Shortest-Path Rerouting

As depicted in Chapter 2, m tunnels of traffic exist between all node pairs in uniform all-to-all independent and identically distributed (I.I.D.) traffic, and each wavelength tunnel has a constant transmission rate R. The size of each transaction L is assumed to be exponentially distributed with the expectation L_0, but any well behaved real-life traffic distribution will yield the same architecture recommendations. The service rate per tunnel per transaction is $\mu = \frac{R}{L_0}$. We previously assume the traffic arrivals form a doubly stochastic Poisson point process with a time-dependent rate of $\lambda(t)$ and provide the detection methods for different changing $\lambda(t)$. In this chapter, we

use a stationary λ to facilitate the demonstration of the queueing system. This can be considered as the real-time short-term value of $\lambda(t)$ at time t.

Sometimes, the incremental traffic between a node pair has to go through other paths from the source node to the destination node, which is called rerouting. Denote the traffic that can be transmitted between the node pair using the shortest path as the primary traffic, and the path as the primary path. The primary path of routing is the shortest path between the source node and the destination node. Denote the traffic needs to be rerouted via other paths as the rerouted traffic and the new paths as the secondary paths. Rerouting can be triggered when the certain queueing delay is reached. Therefore, the schedule holder between each node pair can only hold a certain number of transactions.

The primary path of routing is generated by the shortest-path algorithm and the primary path is the shortest path between the source node and the destination node. In this work, the secondary paths for rerouting are also generated based on the shortest-path algorithm. Apart from the primary path, a rerouting table records all the secondary paths for a node pair in the order of preference generated by the shortest-path algorithm depending on the immediate wavelengths availability. The routing algorithm has been widely studied since the 1990s [13, 41]. The reason to use the shortest path algorithm is its simplicity and the efficiency in both cost and energy [41, 55]. It has been shown in [55] that the shortest-path algorithm is both energy-efficient and cost-efficient with adaptive-to-traffic-demand deployment or lighting-up new fibers.

We use the Petersen graph to demonstrate the rerouting algorithm as shown in Fig. 6-1. The Petersen graph is a Moore graph. Moore graphs and generalized Moore graphs are good network topology candidates for handling the optimal cost efficiency and supporting uniform all-to-all traffic with the least number of wavelengths [24]. For the Petersen graph, the length of the primary path can be 1 hop or 2 hops due to the shortest-path routing. Without loss of generality, we use node pair $A - B$ with the 1-hop primary path to demonstrate. The primary path between node pair $A - B$ is the 1-hop path $A - B$, and the secondary paths can be 4-hop, 5-hop, 7-hop, and

8-hop. If $A-B$ is not available, the next shortest-length path, which is a 4-hop path, will be examined. If all 4-hop paths are blocked, 5-hop paths will examined. If 5-hop paths are blocked, then 7-hop paths are examined and finally 8-hop paths. Figure. 6-2 shows all 4-hop paths and all 5-hop paths between node pair A to B. A full list of secondary paths for rerouting between node pair A to B can be found in Table 1. For a node pair with the 2-hop primary path, the length of secondary paths are from 3-hop to 10-hop. In the following sections, we use the number of hops of a path and the length of a path interchangeably.

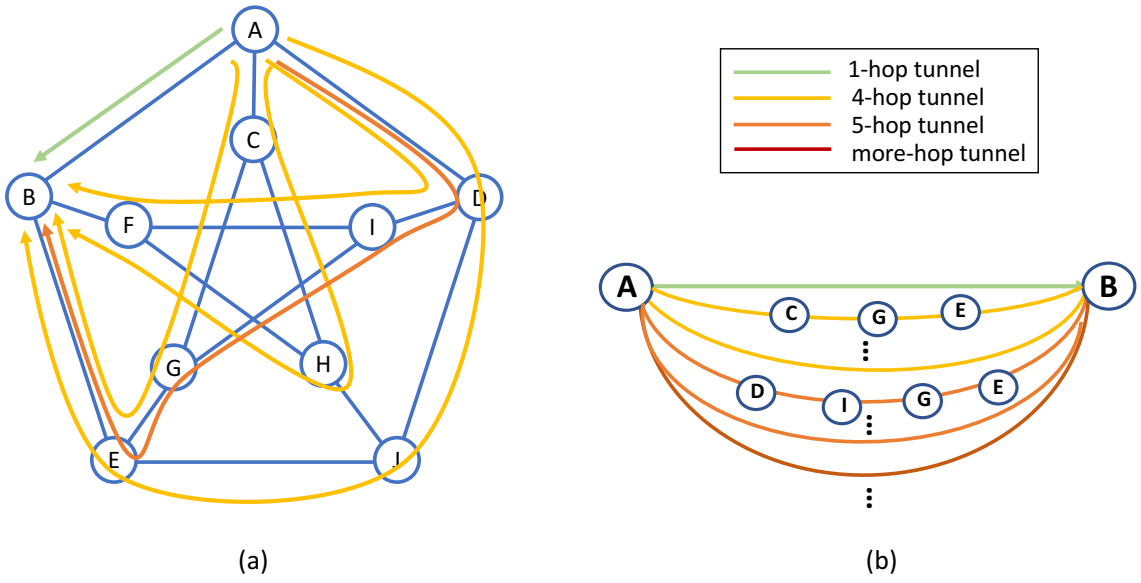

Figure 6-1: The Petersen graph with several secondary paths for rerouting between node pair $A-B$ with 1-hop primary path.

6.2 Blocking of Secondary Paths for Rerouting

The unavailability of wavelengths on one or more hops of a secondary path for rerouting will lead to the blocking of the entire path. When the overall network load is high, the blocking probability of a hop can be high, which reduces the chance of finding an open secondary path, especially for the longer secondary paths. Therefore, we

Table 6.1: Table of secondary paths for rerouting for the node pair A to B in a Petersen graph.

Number of hops	Secondary Path
4	A-C-G-E-B
4	A-C-H-F-B
4	A-D-I-F-B
4	A-D-J-E-B
5	A-C-G-I-F-B
5	A-C-H-J-E-B
5	A-D-I-G-E-B
5	A-D-J-H-F-B
7	A-C-G-E-J-H-F-B
7	A-C-G-I-D-J-E-B
7	A-C-H-F-I-G-E-B
7	A-C-H-J-D-I-F-B
7	A-D-I-F-H-J-E-B
7	A-D-I-G-C-H-F-B
7	A-D-J-E-G-I-F-B
7	A-D-J-H-C-G-E-B
8	A-C-G-E-J-D-I-F-B
8	A-C-G-I-D-J-H-F-B
8	A-C-G-I-F-H-J-E-B
8	A-C-H-F-I-D-J-E-B
8	A-C-H-J-D-I-G-E-B
8	A-C-H-J-E-G-I-F-B
8	A-D-I-F-H-C-G-E-B
8	A-D-I-G-C-H-J-E-B
8	A-D-I-G-E-J-H-F-B
8	A-D-J-E-G-C-H-F-B
8	A-D-J-H-C-G-I-F-B
8	A-D-J-H-F-I-G-E-B

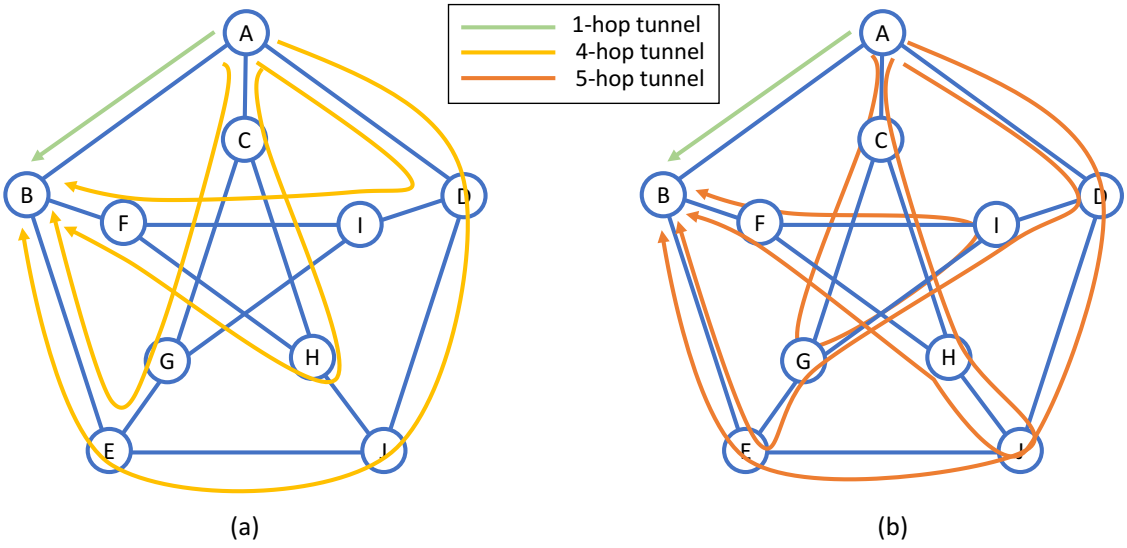

Figure 6-2: Secondary paths for rerouting for the Petersen graph. (a) All the 4-hop paths; (b) All the 5-hop paths.

can further optimize our rerouting algorithm by eliminating the long paths. If all potential short paths are blocked, we need to directly light up new fiber to bring extra wavelengths between the node pair.

6.2.1 Blocking of Long Paths for Rerouting

We analyze the conditional blocking probability of the long secondary paths given that all the short paths are blocked. The blocking probability in optical networks has been widely studied [2, 51, 33, 25, 55, 56]. In most of these works, it is assumed that the blocking of hops is statistically independent of others as an approximation to facilitate the analysis. Similarly, in this work, we assume the blocking on each hop is statistically independent with each other as an approximation and denote the blocking probability of a hop as P_{b_h}. The analysis of the dependent hop-blocking model is proposed in the future work.

Denote the probability that all i-hop secondary paths are blocked as P_{b-i}. The probability of all j-hop paths blocked, given that all i-hop paths are blocked $P_{b-j|b-i}$

is

$$P_{b-j|b-i} = \frac{P_{b-j} \cap P_{b-i}}{P_{b-i}}, \tag{6.1}$$

where $i <= j$ to indicate the order of secondary paths.

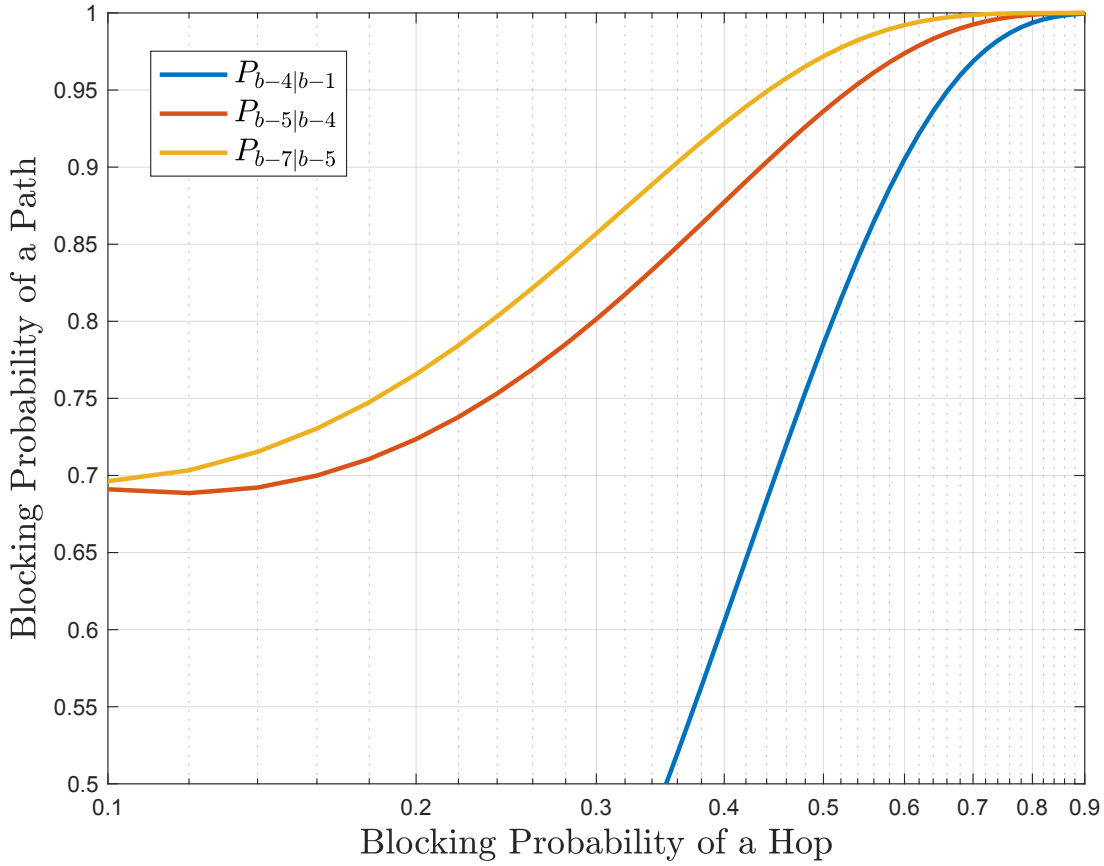

Figure 6-3: Conditional blocking probabilities of different secondary paths on the Petersen graph for the node pair $A - B$ with 1-hop primary path. The blocking on each hop is assumed to be statistically independent of each other.

Figure 6-3 shows the conditional blocking probabilities of long secondary paths given all short secondary paths are blocked on a Petersen graph for node pair $A - B$ with the 1-hop primary path. Here, it is assumed that each hop is statistically

independent of others. If the 1-hop primary path is full, the chance of getting rerouted by a secondary 4-hop path is fairly high. $P_{b-4|b-1}$ is low when P_{b_h} is low due to the independency between all 4-hop paths and the primary path. In contrast, both curves $P_{b-5|b-4}$ and $P_{b-7|b-5}$ start with a high path blocking probability due to the dependency with the previous short paths. There is little need to find a longer secondary path when all 4-hop paths are blocked when P_{b_h} is high. The reduction in blocking probability is insignificant as shown in Fig. 6-3. Similar analysis is valid for node pairs with 2-hop primary paths.

Two reasons are behind the high blocking probability of the long secondary paths. First, more hops mean a higher chance for a path to be blocked, since a path can be blocked if at least one hop in the path is blocked, given the statistical independency among hops. Second, a path with many hops has a high chance of sharing busy hops with the blocked paths. Therefore, if the blocking of hops can be modeled as statistically independent, we make the conjecture that the secondary paths for rerouting are desirable to have two properties: 1. the paths should have small number of hops; 2. the paths should be disjoint with other busy paths. The first property has already been reflected in our rerouting algorithm that implemented with the shortest-path algorithm. The second property provides an improved secondary path ordering for the same-length secondary paths.

6.2.2 Topology Recommendation for Efficient Rerouting

The desirable properties of secondary paths for rerouting provide a hint of the desirable network topology. We examine the path blocking probability on two extreme network topologies as shown Fig. 6-4: 1. the ring topology, where a node only connects to its neighbors; 2. the full mesh topology, where a node connects to every other node. With V nodes in the topology, the ring has the most number of shared paths, and the full mesh has the most number of independent paths. Again, we assume the blocking of hops is statistically independent. We only consider the blocking probability of the shortest secondary paths, since the blocking probability of a long secondary path is high, especially for high-dependency paths.

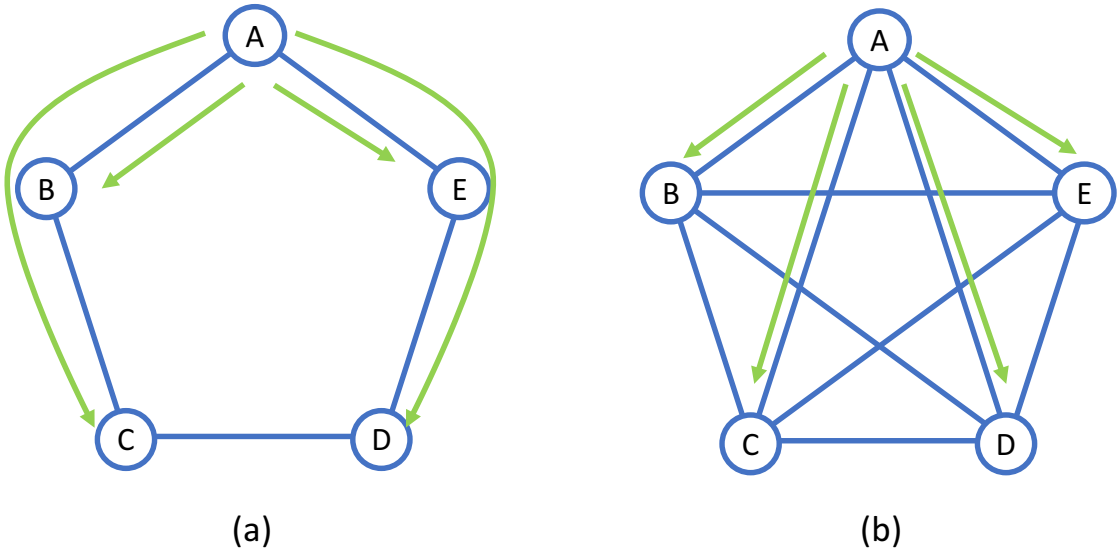

Figure 6-4: Secondary paths for rerouting for two extreme five-node topologies. (a) Ring topology; (b) Full mesh topology.

For a V-node ring topology, there is only one secondary path for each node pair, as a ring can only go either clockwise or counterclockwise. Among all node pairs, the best case (shortest) of a secondary path length is $\lceil \frac{V}{2} \rceil$, where $\lceil \cdot \rceil$ is the ceiling function. With the blocking probability of a hop P_{b_h} and the assumption that all hops are independent, the best case of the probability that all the shortest secondary paths are blocked for a node pair in a V-node ring topology is

$$P_{b_p,r-best} = 1 - (1 - P_{b_h})^{\lceil \frac{V}{2} \rceil}. \tag{6.2}$$

For a V-node ring topology, the worst case (longest) of the secondary path length is $V-1$. With the blocking probability of a hop P_{b_h} and the assumption that all hops are independent, the worst case of the probability that all shortest secondary paths are blocked for a node pair in a V-node ring topology is

$$P_{b_p,r-worst} = 1 - (1 - P_{b_h})^{V-1}. \tag{6.3}$$

For a V-node full mesh topology, there are $V-2$ different 2-hop shortest secondary paths, as the rerouted traffic can go via any of the other $V-2$ nodes from the source to the destination. With the blocking probability of a hop P_{b_h} and the assumption that all hops are independent, the probability that all shortest secondary paths are blocked for a node pair in a V-node full mesh topology is

$$P_{b_p,mesh} = [1-(1-P_{b_h})^2]^{V-2}. \tag{6.4}$$

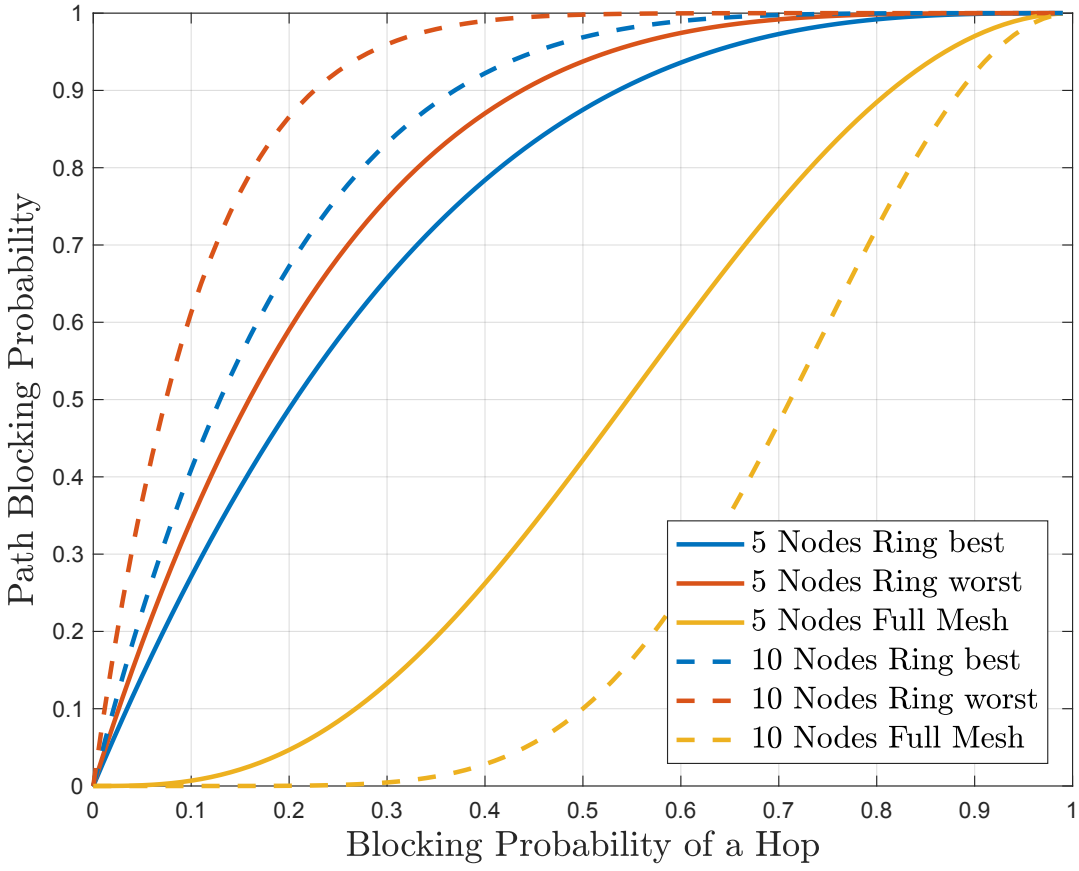

Figure 6-5: Path blocking probability versus the blocking probability on a hop for different topologies. The blocking on each hop is assumed to be statistically independent with each other.

As shown in Fig. 6-5, a full mesh topology has a lower path blocking probability than a ring topology with the same number of nodes given the assumption that all hops are independent. A long path has a high blocking probability as reflected in the ring topology. More independent paths are efficient in reducing the blocking probability as reflected in the full mesh topology. To increase the chance of successful transmission, we make the conjecture that secondary paths are expected to be short and hop-disjoint (edge-disjoint) with the primary path. Both properties require the topology to have good connectivity, which is indicated by the corollary of Menger's Theorem [5] as

Corollary of Menger's Theorem A graph is k-edge-connected if and only if it has at least two nodes and any two nodes can be joined by k edge-disjoint paths.

We can make the conjecture to recommend the topology with high edge-connectivity to increase the rerouting successful rate when the blocking of hop is independent, as the network will have more hop-disjoint paths. Minimum node degree provides an upper bound on the edge-connectivity. That is, if a graph is k-edge-connected, it is necessary that

$$k \leq \sigma(G), \tag{6.5}$$

where $\sigma(G)$ is the minimum degree of any node $v \in V$. Obviously, deleting all edges connecting to a node v will disconnect v from the graph.

6.3 Wavelength Reservation for Rerouted Traffic

A way to improve the successful rate of rerouting is to reserve wavelengths for the rerouted traffic only. As we discussed, shortest-path routing is adopted to find both the primary path and the secondary paths. When the network load is high, reserving wavelength for rerouted traffic can lead to low resource utilization. Assume s wavelengths of traffic can be transmitted between a node pair. If s_r wavelengths

are reserved for the rerouted traffic from other node pairs, only $s - s_r$ wavelengths are available for the primary traffic between the node pair. We can argue that the rerouted traffic will occupy more resources, as the length of the secondary path for rerouting is always the same or longer than the length of the primary path due to the shortest-path algorithm. Define $h_{r_{i,j}}$ as the length of the secondary path for node pair i to j, and $h_{p_{i,j}}$ is the length of the primary path. For a V-node topology with the all-to-all traffic, there are $V(V-1)$ source-destination node pairs. The whole network can support at most $V(V-1)s$ traffic as the theoretical limit with the balanced traffic. Define the traffic supporting ratio α as the ratio of the actual maximum traffic supported to the theoretical limit of the network. For a general V-node topology, the best case of rerouting is bounded by the condition that the network is fully used without any idle wavelengths as

$$\begin{aligned}
\alpha_{best} &\leq \frac{V(V-1)(s-s_r) + \sum\limits_{i,j \in V(G), i \neq j} \frac{s_r h_{p_{i,j}}}{h_{r_{i,j}}}}{V(V-1)s} && (6.6)\\
&= (1 - \frac{s_r}{s}) + \frac{\frac{s_r}{s} \sum\limits_{i,j \in V(G), i \neq j} \frac{h_{p_{i,j}}}{h_{r_{i,j}}}}{V(V-1)} && (6.7)\\
&= 1 - (\frac{s_r}{s}) \left(\frac{1 - \sum\limits_{i,j \in V(G), i \neq j} \frac{h_{p_{i,j}}}{h_{r_{i,j}}}}{V(V-1)} \right), && (6.8)
\end{aligned}$$

where $V(G)$ denotes the nodes set of the V-node topology G. Notice that $\frac{h_{p_{i,j}}}{h_{r_{i,j}}} \leq 1$ since the shortest-path algorithm is used to find both the primary paths and the secondary paths. There are at most $V(V-1)$ terms in the sum, at most because some routes may not be feasible, and there are unused resources in some links. Thus the overall term is less than 1. Hence the inequality indicates an upper bound. The upper bound above is monotonically decreasing in s_r and thus achieves its maximum at the origin.

We can also find a simple upper bound on the α_{best} using the maximum of $\frac{h_{p_{i,j}}}{h_{r_{i,j}}}$ (the shortest secondary path length) as

$$\alpha_{best} \leq \frac{s - s_r + s_r \left(\frac{h_{p_{i,j}}}{h_{r_{i,j}}}\right)_{max}}{s} \triangleq \alpha_{best_{upper}}. \quad (6.9)$$

The worst case of rerouting is that only s_r units of rerouted traffic from a node pair can be fulfilled in the whole network. The secondary path of the rerouted traffic occupies part or even whole secondary paths for other node pairs, and many wavelengths have to be idle. The worst case is

$$\alpha_{worst} = \frac{V(V-1)(s - s_r) + s_r}{V(V-1)s}. \quad (6.10)$$

In Fig.6-6, the region confined by the upper bound of the best case with $h_{r_{i,j}min} = 1.5$ and the worst case indicates the possible traffic supporting ratio range for a 10-node topology, where the shortest-path algorithm is adopted to find both the primary paths and the secondary paths for rerouting. The best case of the Petersen graph is also shown in Fig.6-6, where only the shortest secondary paths are used due to the high blocking probability of the long secondary paths. The wavelength reservation for rerouted traffic sacrifices the traffic supporting ratio for guaranteeing the transmission of some but not all rerouted traffic. In the worst case, a great amount of network resources are wasted as only s_r units of traffic can be supported, compared to totally $V(V-1)s_r$ traffic in the whole network needs to be rerouted when the network is fully loaded($\sim 100\%$). With the increase of the network size, such portion is small, and it can be considered as no rerouting is available.

From the above analysis, reserving wavelength for the rerouted traffic will incur the waste of resources if the shortest-path algorithm is adopted to find both the primary path and the secondary path for rerouting. Reserving wavelength is not recommended if we aim to achieve the high resource utilization. If the wavelength reservation has to be performed to guarantee successful transmission of certain traffic, secondary path planning should be careful to avoid the worst case of the traffic supporting.

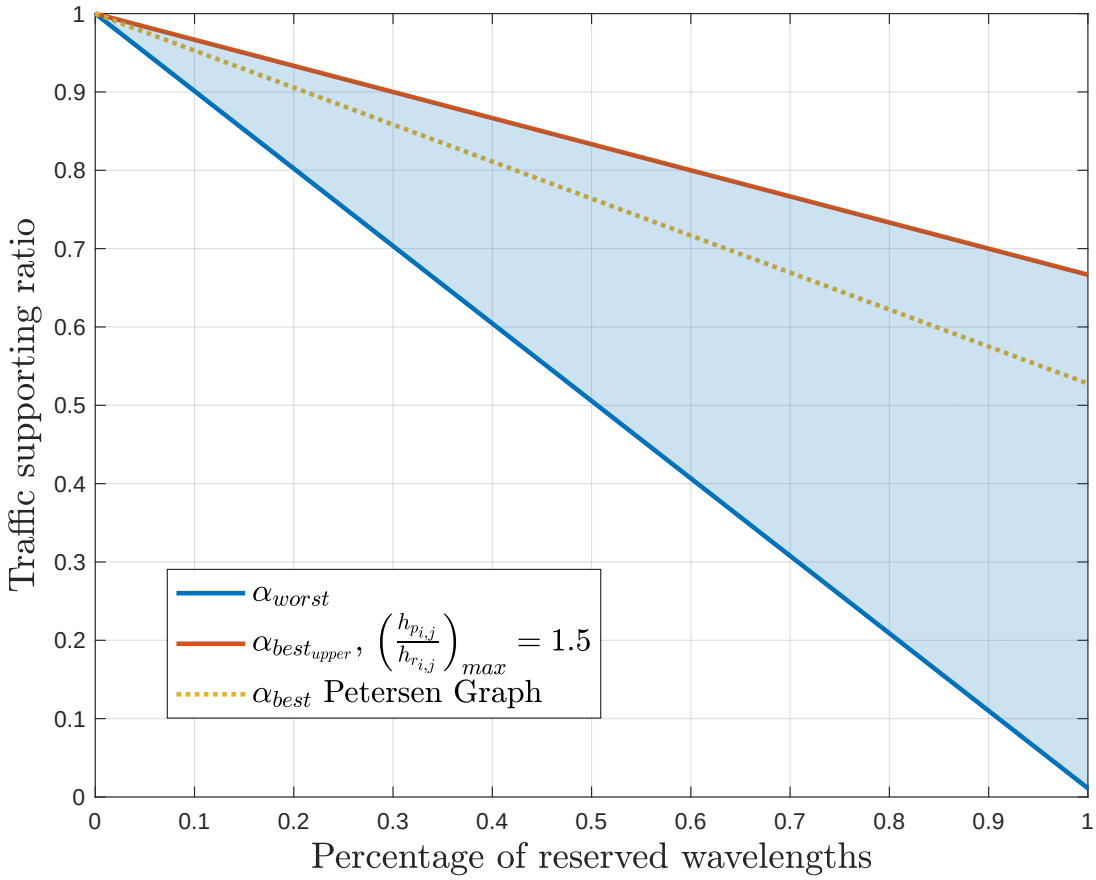

Figure 6-6: The traffic supporting ratio versus percentage of reserved wavelength for a 10-node network topology. The shortest-path algorithm is adopted to find the secondary path for rerouting and only the shortest secondary paths are used.

6.4 Delay Threshold to Reroute

Rerouting can be triggered by a threshold of queueing delay on the primary path, where the queue is first-come-first-serve (FCFS). This rerouting trigger is easier to use than many other more complicated algorithms to save the control efforts. As indicated in [55], a scalable, efficient, and easy-to-implement control plane without compromising too much of the network performance is important in future network design. The queueing delay on the primary path is directly measurable by the queue size and the sizes of the transaction in the queue, which can reflect both the current

network traffic environment and the network configurations.

6.4.1 Triggering Algorithm with the Delay Threshold

Define the queueing delay of the traffic transaction that is going to join the queue as τ_Q. With the queueing delay threshold $\tau_{Q_{Thresh.}}$, the threshold algorithm to trigger rerouting is to reroute when

$$\tau_Q \geq \tau_{Q_{Thresh.}} \tag{6.11}$$

Triggering by $\tau_{Q_{Thresh.}}$ can make a rerouting decision as soon as possible without any predetermined parameters. It is easy to implement as it requires no detail on the traffic arrival patterns.

To facilitate the implementation, we elaborate the triggering algorithm with the queue size and the transmission times of the transactions in the queue. The transmission time of a transaction depends on the size of the transactions. Denote K transactions are in the holder when the rerouting is triggered, and K is a random variable. K can be considered as the maximum number of transactions in the schedule holder, which is determined by the exact traffic transaction sizes in the holder and $\tau_{Q_{Thresh.}}$. Define the transmission time of kth transaction in the holder between node pairs as T_k, $1 \leq k \leq K$. With the fixed transmission rate R and the size of the kth transaction as L_k, T_k is decided by the size of the kth transaction as

$$T_k = \frac{L_k}{R} \tag{6.12}$$

Denote the shortest remaining time of ongoing transmission when the $(K+1)$th transaction is going to join the queue as $\tau_{ongoing}$. With a single wavelength assigned between the node pair, we have

$$\tau_Q = \tau_{ongoing} + \sum_{k=1}^{K} T_k \tag{6.13}$$

With multiple wavelengths assigned between the node pair, τ_Q is the time that the first wavelength is available after all K transactions in the holder enters the transmission as shown in Algorithm 1. K is a random variable depending on the situation when $\tau_{Q_{Thresh.}}$ is crossed and the rerouting is triggered. τ_Q can be calculated case by case with $\tau_{ongoing}$ and T_k, $k = 1, 2, ..., K$. Since the transmission times of all sessions are available and the FCFS schedule is known, the first available wavelength is known at the time of session arrival at the node. Thus, it is a deterministic decision to reroute or not.

Algorithm 3 Rerouting Triggering by $\tau_{Q_{Thresh.}}$ with Multiple Wavelengths Assigned Between the Node Pair

$\tau_Q \leftarrow 0$
$\Delta t \leftarrow \tau_{ongoing}$
for each T_k **do**
  $i \leftarrow$ index of the first open wavelength after Δt
  $\tau_Q \leftarrow \tau_Q + \Delta t$
  Substract Δt from all ongoing transmission times
  Assign the kth traffic transaction to the wavelength i
  $\Delta t \leftarrow$ shortest remaining time of ongoing transmission
end for
$\tau_Q \leftarrow \tau_Q + \Delta t$
if $\tau_Q \geq \tau_{Q_{Thresh.}}$ **then**
  Reroute to the secondary path
else
  Wait in the queue
end if

The triggering by $\tau_{Q_{Thresh.}}$ enables the high resource utilization on the primary

path, as the queue will not accumulate until the lack of wavelengths. In other words, rerouting will not be triggered until the resources on the primary path are fully used. It is consistent with the design of not reserving wavelength for the rerouted traffic for the purpose of high resource utilization.

Traffic in different transaction sizes may have different $\tau_{Q_{Thresh.}}$. Without loss of generality, we highlight two types of traffic as in [26] but on a larger scale:

1. mice traffic with the average size $\sim 10^4$ bits;

2. elephant traffic with the average size ~ 1 Gigabytes.

The size of an elephant traffic transaction is approximately 10^6 times larger than the size of a mouse traffic transaction. To facilitate the analysis, we define a mouse traffic transaction size as a unit in the following analysis. So the size of the an elephant traffic transaction is 10^6 units. With the same transmission rate R, if we define the transmission delay of a mouse traffic transaction as a unit of delay, then the transmission delay of an elephant traffic is 10^6 units of delay.

6.4.2 Performance for Node Pair with Single Wavelength

When a single wavelength is assigned between the node pair, the τ_Q before rerouting is the sum of the transmission times of K transactions and the shortest remaining time of ongoing transmission at time t. Each T_k follows the exponential distribution with the average size L_0, as defined in Chapter 2. Due to the memoryless property of the exponential distribution, $\tau_{ongoing}$ also follows the exponential distribution with the expectation of L_0. Therefore, we can consider $\tau_{ongoing}$ as another T_k, and we denote it as T_0. With a single wavelength assigned between the node pair, we have

$$\tau_Q = \sum_{k=0}^{K} T_k. \tag{6.14}$$

Obviously, τ_Q is therefore a random walk based on T_k as a sequential test to trigger rerouting. Once τ_Q crosses $\tau_{Q_{Thresh.}}$, rerouting will be triggered. K is the random

variable to depict the number of transactions in the holder when the threshold is crossed, which is also the stopping time of the random walk. With Wald's equality, we have the average number of transactions in the holder when the threshold is crossed K with a single wavelength assigned between node pairs as

$$E[\tau_Q] = E[T_k]E[K] \tag{6.15}$$

$$\Leftrightarrow E[K] = \frac{E[\tau_Q]}{E[T_k]} \tag{6.16}$$

$$\Leftrightarrow E[K] = \frac{E[\tau_Q]R}{L_0}. \tag{6.17}$$

Figure 6-7 shows the average number of transactions in the holder when the threshold is crossed K versus the queueing delay threshold $\tau_{Q_{Thresh.}}$ for different traffic situations. The maximum number of transactions in the schedule holder is determined by the exact traffic transaction sizes in the holder and $\tau_{Q_{Thresh.}}$. A large $\tau_{Q_{Thresh.}}$ allows more traffic transactions to stay in the holder, and a small exact transaction size also enables more traffic transactions to stay. The holder is allowed to hold a large number of transactions if all the transactions queueing in the holders are mice traffic, which is the upper bound. In contrast, with the same $\tau_{Q_{Thresh.}}$, the holder is only allowed to hold a small number of transactions if all the transactions are elephants, which is the lower bound. The total number of transactions of different sizes in the holder lies in the shade between the upper bound and the lower bound.

A small $\tau_{Q_{Thresh.}}$ can allow a reasonable number of mice traffic transactions to be held in the holder, but a small $\tau_{Q_{Thresh.}}$ is too strict for the elephant traffic and results in a low number in the holder for the elephant traffic. For example, $\tau_{Q_{Thresh.}} = 10^6$ only allows an elephant to wait in the holder, as shown in Fig. 6-7, and any following incoming elephant traffic has to be rerouted. If $\tau_{Q_{Thresh.}} < 10^6$, no elephant can join the holder if the wavelength is not idle. If both mice and elephant traffic are transmitted together, the same strict $\tau_{Q_{Thresh.}}$ gives the elephant traffic a slight chance to wait in the queue, which impedes the transmission of elephant traffic. As indicated in [29, 30, 26], elephant traffic can only come at an extremely low load when the

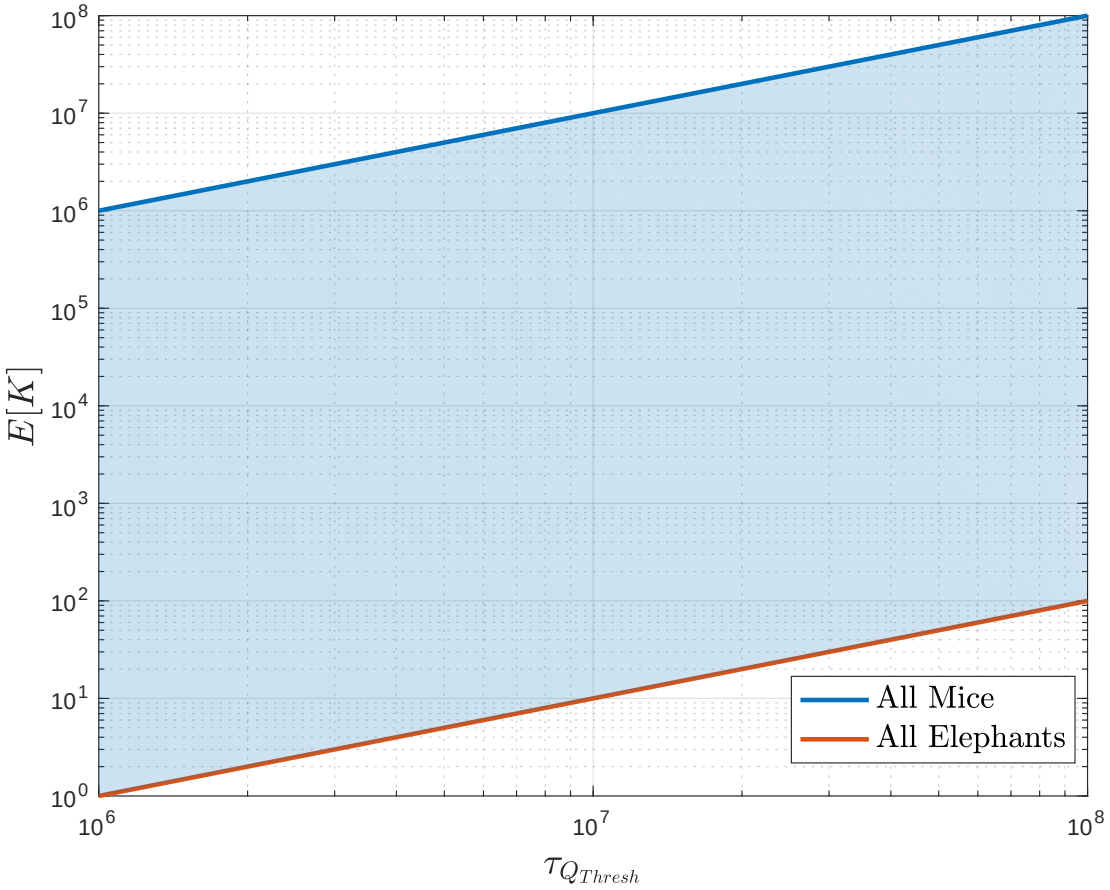

Figure 6-7: The expected number of transactions in the holder when the threshold is crossed $E[K]$ versus the queueing delay threshold $\tau_{Q_{Thresh.}}$ for different traffic situations. The wavelength number is 1 per node pair. A unit delay is defined as the transmission delay of a mouse traffic transaction. The transmission delay of an elephant traffic transaction is 10^6 units of delay.

average queueing delay limit is low. The low operating load will lead to the waste of network resources. It is also shown in [29, 30, 26] that partitioning of resources will improve average normalized delay, as the average delay is the performance metric. Per [26], the optimal wavelength assignment to minimize the average delay is to allocate the wavelengths proportioned by the ratio of average traffic sizes, if the network has the mixed traffic with different average sizes. In our case of using the real-time delay deadline as the metric, we conjecture that the same result will hold. Though it may result in different splits as the metrics in this work and in [26] are different, the idea

of splitting the traffic to relax the delay requirement will hold in both cases.

6.4.3 Performance for Node Pair with Multiple Wavelengths

We further discuss the performance of our queueing delay threshold triggering algorithm when multiple wavelengths are assigned between node pairs. Though the algorithm becomes more complicated than that in the single wavelength situation shown in Algorithm 3, we can find that τ_Q is still a random walk. τ_Q is $\tau_{ongoing}$ plus the sum of every shortest remaining time of ongoing transmission when each transaction in the holder is going to be transmitted. As we argued in the previous section, $\tau_{ongoing}$ follows the exponential distribution due to the memoryless property of the exponential distribution. Therefore, every subsequent shortest remaining time of ongoing transmission is also exponentially distributed. Denote the shortest remaining time of ongoing transmission when the kth transaction in the holder is going to be transmitted as T'_k, and denote $\tau_{ongoing}$ as T'_0. Therefore, we have τ_Q when multiple wavelengths assigned are between the node pairs as

$$\tau_Q = \sum_{k=0}^{K} T'_k. \tag{6.18}$$

As no wavelength reservation is recommended, we assume m wavelengths in total are available between each node pair. Different from the single wavelength assigned between a node pair, more wavelengths assigned can reduce the queueing time as traffic transactions can be transmitted in parallel. With m wavelengths assigned, m transactions can be transmitted at the same time. As stated in the **Corollary of Proposition 1** that we proved in Chapter 3, the average queueing delay for an $M/M/m(t)$ queue with a service rate μ is the same as that of an $M/M/1$ queue with the service rate of $m(t)\mu$. Given the transmission rate for a transaction on the single wavelength as $\mu = \frac{R}{L}$, we can approximately use $m\mu$ as the transmission rate of queue with multiple wavelengths when calculating T'_k. Therefore, with Wald's equality, we have the average number of transactions in the holder when the threshold is crossed

K with m wavelengths assigned between node pairs as

$$E[\tau_Q] = E[T'_k]E[K] \tag{6.19}$$

$$\Leftrightarrow E[K] = \frac{E[\tau_Q]}{E[T'_k]} \tag{6.20}$$

$$\Leftrightarrow E[K] = \frac{E[\tau_Q]mR}{L_0}. \tag{6.21}$$

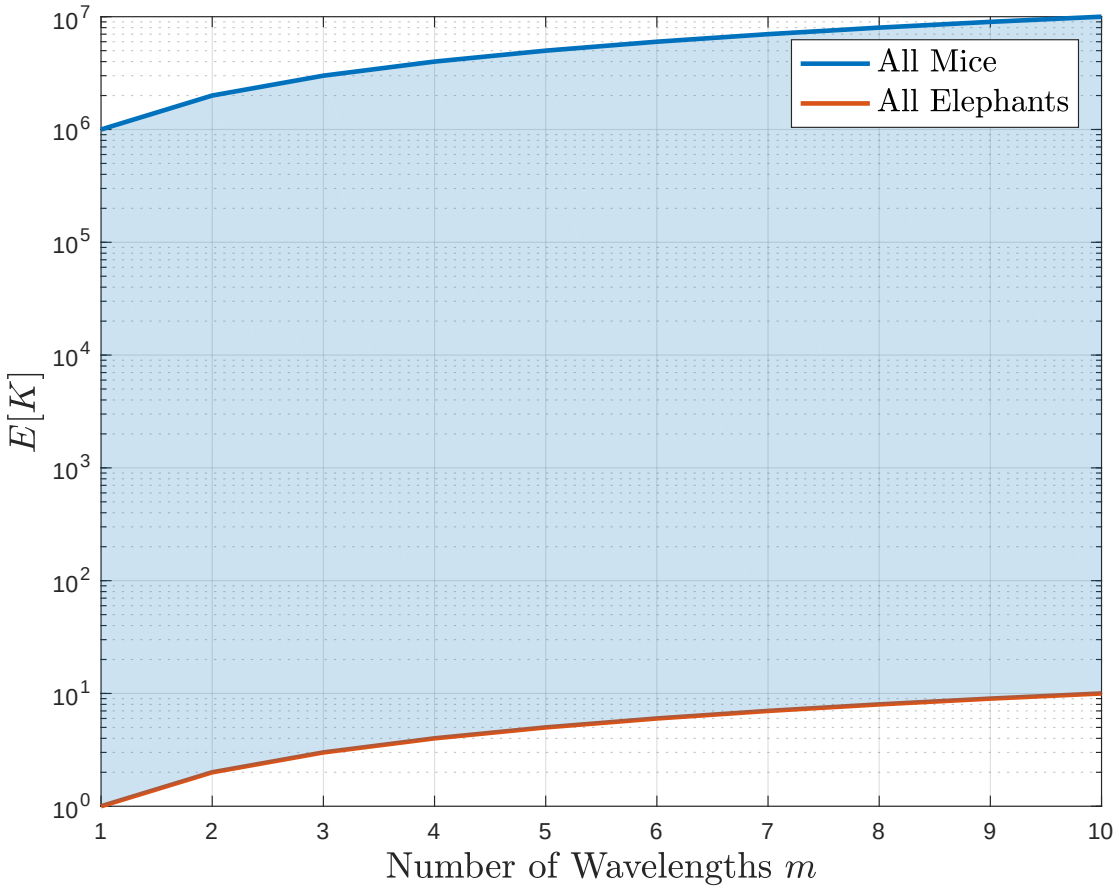

Figure 6-8: The expected number of transactions in the holder when the threshold is crossed $E[K]$ versus the number of wavelengths assigned between each node pair for different traffic situations. A unit delay is defined as the transmission delay of a mouse traffic transaction. The transmission delay of elephant traffic is 10^6 units of delay. $\tau_{Q_{Thresh.}} = 10^6$.

Figure 6-8 shows the number of transactions in the holder when the threshold is crossed K versus the number of wavelengths assigned between each node pair for different traffic situations with $\tau_{Q_{Thresh.}} = 10^6$. The maximum number of transactions in the schedule holder is determined by the exact traffic transaction sizes in the holder and $\tau_{Q_{Thresh.}}$. With more wavelengths assigned between the node pair, the number of traffic transactions that can be held increases. Multiple wavelengths improve the total transmission rate between the node pair, and multiple transactions can be transmitted in parallel.

Similarly, with the same $\tau_{Q_{Thresh.}}$, the schedule holder can hold a large number of mice traffic transactions but only a small number of elephant traffic transactions. The small $\tau_{Q_{Thresh.}}$ is too strict for the elephant traffic and requests the elephant traffic to be rerouted at a low load [29, 30, 26]. There is a trade-off among the queueing delays, load, and blocking probability. A high blocking probability indicates that the network only allows a small number of traffic transactions to be held in the schedule holders on each node pair. Therefore, the traffic have a greater chance to be rerouted rather than waiting in the holder on the primary path, and the queueing delay on the primary path is reduced. With the increase of the load, the queueing delay also increases [29, 30, 26]. If a low $\tau_{Q_{Thresh.}}$ is required, it is reasonable to expect the maximum network load to be low, which may result in low resource utilization. If the requirements on queueing delay and blocking probability are strict, the load must be low and thus will be costly. If we can relax performance requirements, then the network can be handled at high loads.

Splitting the transmission between mice traffic and elephant traffic will solve the problem, as indicated in [29, 30, 26]. A normalized queueing delay requirement should be expected, where the queueing delay is normalized by the transmission time. Mice traffic and elephant traffic usually come from different network services, and they usually have different performance requirements of $\tau_{Q_{Thresh.}}$. To perform the normalized queueing delay requirement, we can split the transmission of the mice traffic and the elephant traffic with a method similar to that in [29, 30, 26]. In [26], Huang recommends partitioning the resources to traffic of different sizes to improve the average

normalized delay, as the average delay is the performance metric. Though the real-time queueing delay is used as the metric in this work, we conjecture that the idea of splitting traffic with different sizes will help to improve the network performance in this work. Splitting the transmission will be a great relief as the absolute $\tau_{Q_{Thresh.}}$ of the elephant traffic will be relaxed, and the network will be able to hold more of it. In other words, more elephant traffic will avoid being rerouted or even blocked. As also shown in [26], the optimal wavelength assignment to minimize the delay is to allocate the wavelengths proportioned by the ratio of average traffic sizes, if the network has the mixed traffic with different average sizes. It may result in different splits in this work as the metrics in this work and in [26] are different.

Therefore, we make the conjecture to recommend splitting the traffic with a great spread of transaction sizes into classes with resources dedicated to the class, as shown in [29, 30, 26]. The delay for any session should be normalized by its transmission time. Notice that the traffic split in [26] is based on the optimized expected delays and is different from the metric in this work, which may result in different detailed splits. It is reasonable to wait a few transmission times for the size of the session (either mice or elephants) and therefore partition the resources. $\tau_{Q_{Thresh.}}$ will be relaxed as larger size traffic are expected to have longer transmission delay. Therefore, the elephant traffic can be transmitted at a high load to save costs. On the other hand, more elephants traffic can be transmitted on the primary path rather than rerouted to secondary paths. This will help to better utilize the network resources, as the length of the primary path is always shorter than the secondary path due to the shortest-path algorithm.

6.5 Summary of Chapter 6

In this chapter, we constructed and discussed a dynamic rerouting scheme. Rerouting is triggered when the queueing delay crosses a threshold, and the threshold is determined by the users' quality of service requirements. Both the primary path for routing and the secondary paths for rerouting will be generated based on the shortest

path algorithm. We make the conjecture that the secondary paths for rerouting are recommended to have as few hops as possible and be disjoint with other busy paths to increase the rate of successful rerouting, if hops are assumed to be statistically independent. The conjecture also suggests a high edge-connectivity topology. With the shortest-path algorithm used to find both the primary path for routing and secondary path for rerouting, we do not recommend reserving wavelengths for rerouted traffic because it yields poor resource utilization. Traffic in different sizes should have different normalized delay deadlines for rerouting. With the same queueing delay threshold, a large traffic transaction has to be rerouted when a very small number of transactions are in the holder, while small traffic transactions can be rerouted when a large number of transactions are held in the holder. We make the conjecture to recommend splitting the traffic into multi-classes and partitioning the resources similarly as in [29, 30, 26]. This will allow the reasonable queueing delay threshold to achieve more frequent use of the primary path before rerouting happens. The delay for any session should be normalized by its transmission time. This is the key to improving resource utilization and making the network affordable.

Chapter 7

Conclusion

In this thesis, we have provided the design of a practical fast-reconfigurable cognitive optical networking management and control scheme. The design aims to quickly catch traffic changes and perform rapid adaptations to maintain a low queueing delay without large over-provisioning.

Based on the simplified characteristics of dynamic, bursty, and high-granularity traffic, we have built a simplified traffic model where the traffic arrival rate changes follow a multi-state Markov process. The traffic is categorized into different network traffic environments by the length of the network coherence time, which is the time that it takes the traffic to change. An end-to-end tunneled network architecture is used to reduce the control complexity.

We have addressed the design of a fast-response algorithm for wavelength reconfiguration in the long network coherence time environment. Two Bayesian estimators and a stopping-trial sequential estimator are further elaborated from [12] and examined to detect changes of traffic arrival statistics. Based on the transient behaviors of the network, we have shown that the stopping-trial estimator has the shortest detection time for traffic rate changes, and it requires no knowledge of *a priori* probabilities, except the inter-arrival times of the sessions have to be independent. With continuous assessment, the system reconfigures only when it is necessary. Allowing for the possibility of the addition and subtraction of multiple wavelengths, the stopping-trial estimator (among all three estimators) requires the smallest number of wavelengths

to be reconfigured due to its fastest response that helps to avoid a high peak queueing delay.

In the moderate network coherence time environment and the short network coherence time environment, we have shown that the stopping-trial estimator can still provide the desired wavelength assignment scheme. The stopping-trial estimator can make reconfiguration decisions in the shortest possible time with independent but not necessarily identically distributed inter-arrival times of traffic sessions. The beauty of the stopping-trial estimator is that it can trigger reconfigurations without having to infer or estimate the network traffic statistics.

To deal with the super-fast traffic rate changes, we have modeled the transient behavior of the network traffic drifts towards convergence to a new steady state analytically to validate the feasibility of the algorithm to predict traffic changes. As an example of a specific network drift, we have provided a traffic trend estimation technique for the fast traffic rate changes approximated by a simple linear model. With a few arrivals, the sequential maximum likelihood estimator based on distribution of the inhomogeneous Poisson process can estimate the network traffic drifting trend and enable reconfigurations in advance to minimize the buildup of congestion. The algorithm makes network management and control more efficient to avoid large over-provisioning and yields more affordable architecture in dynamic environments to meet the requirement of future dynamic traffic.

When increasing traffic can not be fully handled by the fast wavelength reconfiguration, we have constructed a cognitive dynamic rerouting scheme. We have designed rerouting that is triggered when the queueing delay crosses a threshold, and the threshold is determined by users' quality of service requirements. Both the primary path for routing and the secondary paths for rerouting will be generated based on the shortest-path algorithm. We have made the conjecture that the secondary paths for rerouting are recommended to have as few hops as possible and to be disjoint with other busy paths to increase the successful rerouting rate, when hops on the paths are assumed statistically independent, which is simplification. The conjecture also suggests that a high edge-connectivity topology should be used. We do not recommend

reserving wavelengths for rerouted traffic because it yields poor resource utilization when the shortest-path algorithm is used to find both the primary path for routing and the secondary paths for rerouting. With the same queueing delay threshold, the queue (the schedule holder) can only accept a small number of large traffic transactions, while a large number of small traffic transactions can be be held in the holder. As is shown in [29, 30, 26], partitioning of resources will improve average normalized delay. Based on this and our analysis of the number of transactions in the schedule holder, we make the conjecture to recommend splitting the traffic into multi-classes and partitioning the resources and normalize the queueing delay for any session by its transmission time. The reasonable normalized queueing delay threshold can increase the number of transactions that can be held in the schedule holders before rerouting occurs. This will allow the most frequent use of the primary path and is the key to improving resource utilization and making the network affordable.

Possible future work is as follows:

1. In our non-stationary traffic model, we assume a doubly stochastic Poisson process with changing arrival rates. When the actual traffic distribution has a heavy tail, which encourages more frequent arrivals of large traffic, one could study to what extent our algorithms still provide good performance in both detection and reconfiguration. Besides, we assume the change of the traffic arrival rates follows a Markov process and also provide the traffic trend estimation in the fast change environment if the arrival rate changes in a linear model. One could also study the performance of other arrival rate change patterns.

2. In our rerouting algorithm design, we make the conjecture that the dependencies among paths will affect the availability of the long secondary paths. One could study how the dependency among the hops affects the availability of the secondary paths analytically. Besides, we assume the blocking of a hop is independent with each other. In many other scenarios, the availability of hops on a path can be partially correlated or even completely correlated. One could study the blocking of the secondary paths given that the blocking of the hop is

not independent.

3. We adopt a queueing delay threshold to trigger the rerouting. A more comprehensive analysis of its performance should be conducted. One could study the impacts of different queueing delay thresholds on the allowable network load and the blocking probability. Since this is a heuristic algorithm, a performance comparison with other rerouting triggering mechanisms should be also studied.

Bibliography

[1] M Agaskar. *Architectural Constructs for Time-Critical Networking in the Smart City*. PhD thesis, Massachusetts Institute of Technology, 2018.

[2] R. A. Barry and P. A. Humblet. Models of blocking probability in all-optical networks with and without wavelength changers. *IEEE Journal on Selected Areas in Communications*, 14(5):858–867, 1996.

[3] D.P. Bertsekas and R.G. Gallager. *Data Networks (2Nd Ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, 1992.

[4] G. Blom. A generalization of wald's fundamental identity. In *Ann. Math. Statist.*, volume 20, pages 439–444, 1949.

[5] B. Bollobas. *Modern Graph Theory*. Springer-Verlag Now York Inc., New York, NY, 1998.

[6] R. Borkowski, R. J. Duran, C. Kachris, D. Siracusa, A. Caballero, N. Fernandez, D. Klonidis, A. Francescon, T. Jimenez, J. C. Aguado, I. de Miguel, E. Salvadori, I. Tomkos, R. M. Lorenzo, and I. T. Monroy. Cognitive optical network testbed: Eu project chron. *IEEE/OSA Journal of Optical Communications and Networking*, 7(2):A344–A355, February 2015.

[7] V. W. S. Chan. Cognitive optical wireless network. In *2016 18th International Conference on Transparent Optical Networks (ICTON)*, pages 1–4, 2016.

[8] V. W. S. Chan. Scalable and dynamic optical network architecture. In *2016 21st OptoElectronics and Communications Conference (OECC) held jointly with 2016 International Conference on Photonics in Switching (PS)*, pages 1–3, 2016.

[9] V. W. S. Chan. Resilient optical networks. In *2018 20th International Conference on Transparent Optical Networks (ICTON)*, pages 1–4, 2018.

[10] V. W. S. Chan and E. Jang. Cognitive all-optical fiber network architecture. In *2017 19th International Conference on Transparent Optical Networks (ICTON)*, pages 1–4, 2017.

[11] V.W.S. Chan. Optical flow switching networks. *Proceedings of the IEEE*, 100(5):1079–1091, May 2012.

[12] V.W.S. Chan. Cognitive optical networks. In *International Conference on Communications(ICC)*, pages 1–6, May 2018.

[13] I. Chlamtac, A. Ganz, and G. Karmi. Lightnet: lightpath based solutions for wide bandwidth wans. In *Proceedings. IEEE INFOCOM '90: Ninth Annual Joint Conference of the IEEE Computer and Communications SocietiesThe Multiple Facets of Integration*, pages 1014–1021 vol.3, 1990.

[14] CHRON. Cognitive heterogeneous reconfigurable optical network. `https://cordis.europa.eu/project/id/258644l`. Accessed: 2020-05-12.

[15] Cisco. Cisco annual internet report (2018 – 2023) white paper. `https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html`. Accessed: 2020-05-06.

[16] Cisco. Cisco visual networking index: Forecast and trends, 2017 – 2022 white paper. `https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html`. Accessed: 2019-11-17.

[17] J. Clark. *Estimation for Poisson processes with applications in optical communication*. PhD thesis, Massachusetts Institute of Technology, 1971.

[18] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, New York, NY, USA, 2006.

[19] I. de Miguel, R. J. Durán, T. Jiménez, N. Fernández, J. C. Aguado, R. M. Lorenzo, A. Caballero, I. T. Monroy, Y. Ye, A. Tymecki, I. Tomkos, M. Angelou, D. Klonidis, A. Francescon, D. Siracusa, and E. Salvadori. Cognitive dynamic optical networks [invited]. *IEEE/OSA Journal of Optical Communications and Networking*, 5(10):A107–A118, Oct 2013.

[20] R. J. Durán, I. De Miguel, D. Sánchez, N. Fernández, T. Jiménez, J. C. Aguado, V. K. Yedugundla, M. Angelou, N. Merayo, P. Fernández, N. Atallah, R. M. Lorenzo, A. Francescon, I. Tomkos, and E. J. Abril. A cognitive decision system for heterogeneous reconfigurable optical networks. In *2012 Future Network Mobile Summit (FutureNetw)*, pages 1–9, July 2012.

[21] R.G. Gallager. *Stochastic Processes: Theory for Applications*. Cambridge University Press, New York, NY, 2014.

[22] Google. Video ai – enable powerful content discovery and engaging video experiences. `https://cloud.google.com/video-intelligence`. Accessed: 2020-05-06.

[23] Albert Greenberg, James Hamilton, David A. Maltz, and Parveen Patel. The cost of a cloud: Research problems in data center networks. *SIGCOMM Comput. Commun. Rev.*, 39(1):68?73, December 2009.

[24] K.C. Guan. *Cost-effective Optical Network Architecture: A Joint Optimization of Topology, Switching, Routing and Wavelength Assignment*. PhD thesis, Massachusetts Institute of Technology, 2007.

[25] Guanglei Liu, Chuanyi Ji, and V. W. S. Chan. On the scalability of network management information for inter-domain light-path assessment. *IEEE/ACM Transactions on Networking*, 13(1):160–172, 2005.

[26] H Huang. *Hybrid Flow Data Center Network Architecture Design and Analysis*. PhD thesis, Massachusetts Institute of Technology, 2017.

[27] H. Huang and V. W. S. Chan. Transport layer protocol for optical flow-switched networks. In *2013 IEEE International Conference on Communications (ICC)*, pages 3819–3824, 2013.

[28] H. Huang and V. W. S. Chan. Optical flow-switched transport layer protocol design and performance analysis. *IEEE/OSA Journal of Optical Communications and Networking*, 6(9):801–815, 2014.

[29] H. Huang and V. W. S. Chan. Data center optical data link layer architecture design and analysis. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–7, 2016.

[30] H. Huang and V. W. S. Chan. Dynamic optical data center network load balancing and resource allocation. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, 2016.

[31] E. Jang. *Characterization and Performance Analysis of a Cognitive Routing Scheme for a Metropolitan-Area Sensor Network*. Master's thesis, Massachusetts Institute of Technology, 2016.

[32] L. Kleinrock. *Queueing Systems*, volume I: Theory. Wiley Interscience, 1975. (Published in Russian, 1979. Published in Japanese, 1979. Published in Hungarian, 1979. Published in Italian 1992.).

[33] G. Liu, C. Ji, and V. Chan. Network management information for light-path assessment: trade-off between performance and complexity. In *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, volume 2, pages 1362–1372 vol.2, 2003.

[34] B. S. Manoj, R. R. Rao, and M. Zorzi. Cognet: a cognitive complete knowledge network system. *IEEE Wireless Communications*, 15(6):81–88, 2008.

[35] J. Mitola. *Cognitive Radio – An Integrated Agent Architecture for Software Defined Radio*. PhD thesis, KTH Royal Institute of Technology, 2000.

[36] I. T. Monroy, D. Zibar, N. G. Gonzalez, and R. Borkowski. Cognitive heterogeneous reconfigurable optical networks (chron): Enabling technologies and techniques. In *2011 13th International Conference on Transparent Optical Networks*, pages 1–4, June 2011.

[37] P.M. Morse. *Queues, Inventories and Maintenance: The Analysis of Operational Systems with Variable Demand and Supply*. Publications in operations research. John Wiley & Sons, 1965.

[38] A. Rezaee and V. W. S. Chan. Cognitive network management and control with significantly reduced state sensing. *IEEE Transactions on Cognitive Communications and Networking*, 5(3):783–794, 2019.

[39] T.L. Saaty. *Elements of queueing theory: with applications*. McGraw-Hill, 1961.

[40] Y. Shao, A. Rezaee, S. C. Liew, and V. W. S. Chan. Significant sampling for shortest path routing: A deep reinforcement learning solution. In *2019 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, 2019.

[41] J. M. Simmons. *Optical network design and planning*. Springer, New York, NY, 2008.

[42] D. Siracusa, A. Broglio, A. Francescon, A. Zanardi, and E. Salvadori. Toward a control and management system enabling cognitive optical networks. In *Proceedings of the 2013 18th European Conference on Network and Optical Communications 2013 8th Conference on Optical Cabling and Infrastructure (NOC-OC I)*, pages 233–240, 2013.

[43] D. Siracusa, E. Salvadori, A. Francescon, A. Zanardi, M. Angelou, D. Klonidis, I. Tomkos, D. Sánchez, R. J. Durán, and I. de Miguel. A control plane framework for future cognitive heterogeneous optical networks. In *2012 14th International Conference on Transparent Optical Networks (ICTON)*, pages 1–4, 2012.

[44] Sedona Systems. The emerging control hierarchy for service provider sdn-enabled networks. `https://sedonasys.com/2018/01/23/the-emerging-control-hierarchy-for-service-provider-sdn-enabled-networks/`. Accessed: 2020-05-12.

[45] R. W. Thomas, L. A. DaSilva, and A. B. MacKenzie. Cognitive networks. In *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005.*, pages 352–360, Nov 2005.

[46] Ryan W. Thomas, Daniel H. Friend, Luiz A. DaSilva, and Allen B. MacKenzie. *Cognitive Networks*, pages 17–41. Springer Netherlands, Dordrecht, 2007.

[47] J.P. Tignol. *Galois' Theory of Algebraic Equations: Second Edition.* World Scientific Publishing Company Pte Limited, 2015.

[48] W. Wei, C. Wang, and J. Yu. Cognitive optical networks: key drivers, enabling techniques, and adaptive bandwidth services. *IEEE Communications Magazine*, 50(1):106–113, January 2012.

[49] G. Weichenberg, V.W.S. Chan, and M. Medard. Design and analysis of optical flow-switched networks. *Optical Communications and Networking, IEEE/OSA Journal of*, 1(3):B81–B97, August 2009.

[50] Y. Wen. *Scalable Fault Management Architecture for Dynamic Optical Networks: An Information-Theoretic Approach.* PhD thesis, Massachusetts Institute of Technology, 2008.

[51] Yuhong Zhu, G. N. Rouskas, and H. G. Perros. A path decomposition approach for computing blocking probabilities in wavelength-routing networks. *IEEE/ACM Transactions on Networking*, 8(6):747–762, 2000.

[52] G. Zervas, K. Banias, B. R. Rofoee, N. Amaya, and D. Simeonidou. Multi-core, multi-band and multi-dimensional cognitive optical networks: An architecture on demand approach. In *2012 14th International Conference on Transparent Optical Networks (ICTON)*, pages 1–4, July 2012.

[53] G. S. Zervas and D. Simeonidou. Cognitive optical networks: Need, requirements and architecture. In *2010 12th International Conference on Transparent Optical Networks*, pages 1–4, June 2010.

[54] L. Zhang. *Fast scheduling for Optical Flow Switching.* Master's thesis, Massachusetts Institute of Technology, 2010.

[55] L. Zhang. *Network Management and Control of Flow-Switched Optical Networks: Joint Architecture Design and Analysis of Control Plane and Data Plane with Physical-Layer Impairments.* PhD thesis, Massachusetts Institute of Technology, 2014.

[56] L. Zhang and V. Chan. Joint architecture of data and control planes for optical flow switched networks. In *2014 IEEE International Conference on Communications (ICC)*, pages 3425–3431, 2014.

[57] A. X. Zheng, L. Zhang, and V. W. S. Chan. Metropolitan area network architecture for optical flow switching. *IEEE/OSA Journal of Optical Communications and Networking*, 9(6):511–523, 2017.

[58] A.X. Zheng. *Metropolitan Area Network Architecture Design for Optical Flow Switching.* Master's thesis, Massachusetts Institute of Technology, 2015.

[59] A.X. Zheng and V.W.S. Chan. Cognitive management and control for wavelength assignment and reconfiguration in optical networks. In *International Conference on Communications(ICC)*, pages 1–7, May 2019.

[60] A.X. Zheng and V.W.S. Chan. Cognitive management and control of optical networks in dynamic environments. In *International Conference on Communications(ICC)*, pages 1–7, June 2020.