Fast Scheduling for Optical Flow Switching

by

Lei Zhang

B.Eng., Nanyang Technological University (2007)

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

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2010

Author .....
Department of Electrical Engineering and Computer Science
May 7, 2010

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

Accepted by .....
Terry P. Orlando
Chairman, Department Committee on Graduate Students

Fast Scheduling for Optical Flow Switching

by

Lei Zhang

Submitted to the Department of Electrical Engineering and Computer Science
on May 7, 2010, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

Optical Flow Switching (OFS) is a key enabler of future scalable optical networks. In the past decade, the OFS architecture has been studied to build an all-optical data plane to provide an end-to-end, cost-effective data transport to users with large transactions. Flow switching provides low-cost service to high-end users by relieving the IP routers on the edge of wide area networks from large transactions. However, the scheduling process of OFS presents possible queuing delays of several transaction durations. For some special applications with urgent time deadlines, the users want to bypass the queuing and pay more to use the network as soon as possible.

In this thesis, we propose a fast scheduling algorithm which utilizes a probing approach to enable OFS to set up end-to-end connections for users with urgent transactions with a delay of slightly more than one round-trip time. A central control manager is used to periodically collect from network regions their most recent entropy evolutions of the network states and broadcast this information across the whole network in the control plane. With this information, fast setups of end-to-end all-optical connections for OFS are achieved by probing independent paths between source and destination, and reserving the available light paths along the way. A modified Bellman-Ford algorithm is designed to select the paths with the least blocking probabilities. By grouping details of network states into the average entropy, we can greatly reduce the information collected and disseminated by the centralized controller, making the network management and control system scalable to large networks.

Since our algorithm makes no assumptions about network models or traffic statistics, it is robust against model variations, and any future changes in network topologies and traffic patterns. The technique can also be used in heterogeneous networks, in which networks from different domains are interconnected to provide a broader coverage.

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

Acknowledgments

My ultimate gratitude goes to my research adviser, Professor Vincent W.S. Chan, for his expertise, patience, and most of all, for his kindness. Without his support and encouragement, this thesis would not have been possible.

I appreciate NSF, DARPA, and Cisco for the financial support to this research.

I would like to thank all these people who have helped me through the process of the thesis. Specially, I would like to thank Professor Robert G. Gallager for his insightful and inspiring discussions. My thanks go to my officemates Guy Weichenberg, Andrew Puryear, and my friend Yehua Wei for their help on understanding the problem and simulations. I am also thankful to the research scientist Dr. John Chapin in our group and my English writing teacher Jane Dunphy, for their help on how to phrase the technical details into concise and formal writings, and to my friend Yiyang Pei, for her help on improving my Latex skills, which makes the thesis writing a smoother process.

I am fortunate to have Etty Lee, Mia Yinuo Qian, James Glettler, Anurupa Ganguly, Rui Li, Matthew Carey and Katherine Lin as fellow graduate students in our group. They have made my life in Claude E. Shannon Communication and Network Group colorful and enjoyable.

Lastly, I would like to thank my parents, Wenguo Zhang and Chunling Jiang, for their unconditional love and support. This thesis is dedicated to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the invention and development of optical fibers in the 1970s, they have been used as a replacement for copper links to provide a tremendous amount of bandwidth, about 30 THz per fiber. However, as the network architecture is still optimized for traditional electronics communications and switching, the achievable capacity of optical networks is constrained by the speed of electronics at network nodes and far less than fully-utilized. Since the 2000s, driven by exponential growth in bandwidth demand, enabled by advancement of optical network devices like wavelength division multiplexer (WDM), Erbium-doped fiber amplifier (EDFA), optical cross connect (OXC), etc, in conjunction with Generalized Multi-Protocol Label Switching (GMPLS), traffic going though the wide area network (WAN) has been enabled to bypass the expensive electronic core routers and instead use optical tunnels through the WAN [25]. Nevertheless, users are still restrained from access to the vast bandwidth provided by the optical backbone network because of the use of electronically switched metropolitan area network (MAN) and access network. To realize the high rate and low cost of optical networks, a new architectural design that takes full advantage of the properties of optical devices is necessary.

In [1, 4–6, 8–10, 14, 18–25], the authors have proposed and explored a new optical transport mechanism, *Optical Flow Switching* (OFS), to provide end-users with direct and cost-effective access to the core network bandwidth. To efficiently use network resources and make the network management and control system scalable, service

holding times of wavelength channels are required to be on the order of hundreds of milliseconds or longer. Small transactions are assumed to be sent via conventional IP packet switching. All flows go though the schedulers to request for transmission and will be held from transmission until the network is free for the transaction. Scheduling leads to high network utilization albeit with some queuing delay. However, there are some special applications that have tight time deadlines and will not like to go through the normal scheduling process which may result in a transmission delay of several transaction times. Thus, there are users that are willing to pay more to use the network as soon as possible.

The demand of fast transport of large transactions with low delay calls for a new flow switching algorithm that bypasses per flow scheduling but still obtains a clear connection with high probability. In [7], the authors proposed a new fast connection-setup algorithm for OFS which utilizes a probing approach with little more than one round-trip delay time. To differentiate the new algorithm from the normal scheduling algorithm of OFS, which utilizes schedulers on the edges of WAN, we call it "fast scheduling". Their "fast scheduling" algorithm has the drawback that the network management system needs to periodically broadcast the states of all links. Moreover, they assumed a particular traffic model, homogeneous Poisson traffic arrival and exponential or deterministic departure processes. In this thesis, we propose a vastly simplified algorithm which will make "fast scheduling" robust to traffic models, and scalable to large networks and heterogeneous networks with multiple administrative domains. [27]

1.1 Optical Flow Switching

OFS is a key enabler of future scalable optical networks [10]. It was first introduced in [1]. It is a scheduled, end-to-end transport service, in which user-to-user all-optical connections are set up prior to transactions upon end users' requests, to provide them with cost-effective access to the core network bandwidth [8, 12, 20–23]. In particular, OFS is intended for users with large transactions, which are expected to increasingly

contribute to future traffic volume [25]. In addition to improving the quality of service of its direct users, OFS also lowers access costs for other users by relieving all network routers from serving large transactions. OFS can be readily implemented through today's device technologies [15].

1.1.1 Scheduling in Optical Flow Switching

OFS provides a scheduled, all-optical, end-to-end transport service to users with large transactions. To achieve scalable network management and control, service holding times of light paths in OFS are required to be on the order of hundreds of milliseconds or longer [14,20]. Upon user's flow-based request, end-to-end connections are established by negotiations between the schedulers at both the ingress and egress MANs. The user is held from transmission until an all-optical end-to-end light path connection is available.

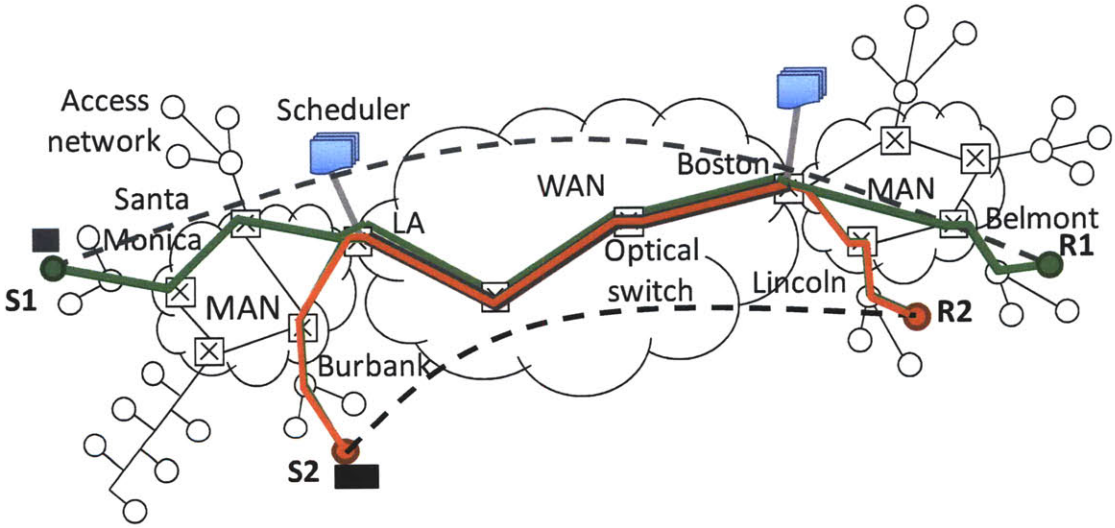

Figure 1-1: A schematic diagram of the scheduling of transactions for Optical Flow Switching. [17]

Figure 1-1 shows an example of two users' scheduled transactions across the WAN without colliding with each other. As shown in the figure, user S1 in Santa Monica wants to send a large data file to user R1 in Belmont; user S2 in Burbank wants to send a large data file to user R2 in Lincoln. Both S1 and S2 send their requests of

transmissions to the scheduling node at LA. The LA scheduler then negotiates with the Boston scheduler on times, routes, and wavelength channels for the transmissions. When decisions are made, the LA and Boston schedulers notify S1/R1 and S2/R2, respectively, of the results of the negotiation. At the agreed times, all-optical, end-to-end connections are set up, and transmissions follow immediately. The scheduling resolves contention and prevents collision, resulting in high link utilization. [17]

1.1.2 Probing in Optical Flow Switching

Since in OFS all flows go through the MAN schedulers to request transmissions and are held until the network is ready, these procedures normally present queuing delays via admission control at the senders. Some special applications, however, have tight time deadlines and are willing to pay more to gain immediate access. Examples that use OFS as a fast transport could be grid computing, cloud computing, and distributed sensor data ingestion. The demand of fast transport of large transactions with low delays calls for a new flow switching algorithm that bypasses scheduling but still obtains a clear connection with high enough probability.

In [7], the authors designed and analyzed fast scheduling algorithms for OFS that meet setup times only slightly longer than one round-trip time. The connection is set up by probing independent candidate paths as announced periodically by the scheduler from source to destination and reserving the available paths along the way. To make the analysis of determining the number of paths to probe tractable, they assumed statistical models of homogeneous Poisson traffic arrival and exponentially distributed departure processes for all the paths connecting source and destination. This is unrealistic and not robust to model variations. For the heterogeneous traffic case, they assumed complete statistics of every link are updated periodically in the control plane. This control traffic itself can be large ($\sim$32 Gbps) in a network of the size of the US backbone network [7]. In addition, their algorithms are highly dependent on the statistical models of arrivals and service times.

1.2 Probing-Enabled Fast Scheduling for Optical Flow Switching

In this work, we have designed a fast scheduling algorithm for OFS which also utilizes the probing approach but does not depend on the assumptions of the statistics of the traffic. As shown in Figure 1-2, a central network manager is used in the control plane to periodically broadcast the set of available links and the information about the evolution of the network states to all nodes. To reduce the amount of state information that has to be sensed and disseminated, the evolution of the network state of each network domain is grouped into one measurable parameter: the average entropy. We chose entropy as the metric because entropy is a good measure of uncertainties. In particular, the higher the entropy, the less certain we are about the state of the network. In addition, average mutual information is used to capture the correlation between neighboring links. Higher mutual information reflects higher correlation between neighboring links. On the other hand, since we only know the average entropy of a group of paths, instead of the detailed blocking probability of each individual path, the paths in one group are considered as having a distribution of blocking probabilities consistent with the reported average entropy. The blocking probability of each path in one group is considered as an independently and identically distributed random variable in our problem formulation. This modeling makes sense, since we do not assume to have a detailed model of the traffic and the blocking probabilities of the paths are not sensed and disseminated.

To capture an available light path for transmission, the algorithm employs a distributed approach in which the source node sends out probing signals along multiple pre-computed paths, with assistance of the broadcast information from the central network manager, to detect and reserve the available paths. We do not want to probe too many paths as the amount of reservation traffic will be too high; we may also end up reserving more than enough resources while taking them out of the pool of available light paths during the booking period. Thus, we want to probe just enough paths to achieve a certain target blocking probability.

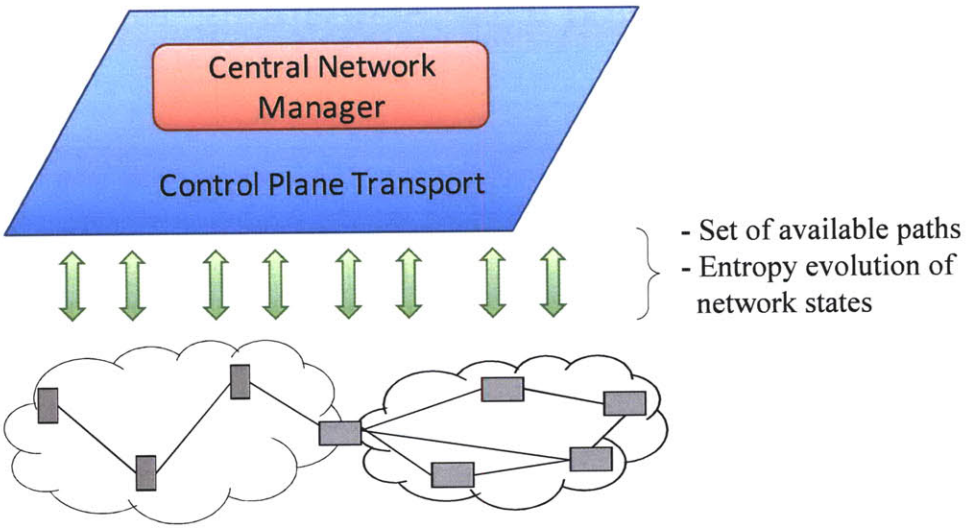

Figure 1-2: A schematic diagram of the relationship between the control plane and networks. In the control plane transport, the set of available paths and entropy evolutions are broadcast to each node in the network.

1.2.1 Broadcasts at Two Time Scales

In the control plane transport, time evolutions of entropy and mutual information and the set of available paths are broadcast periodically at two different time scales as shown in Figure 1-3. The evolutions of entropy (and mutual information) are broadcast with a period of their coherence time (that can range from several minutes to several hours, depending on the actual traffic statistics), whereas the set of available paths is broadcast with a period of 0.3 to 0.5 of the average transaction time ($\geq 1S$). With the evolution of entropy (and mutual information), we can get a close approximation of the average entropy (and mutual information) at any time in the next interval between state broadcasts.

1.2.2 The Probing Approach in Our Algorithm

The probing approach in our algorithm is similar to the one in [7]. When there is a transaction to be made, the source node calculates the number of paths to probe based on the broadcast information about the network states. It randomly selects the calculated number of paths from the set of available paths and sends out probing

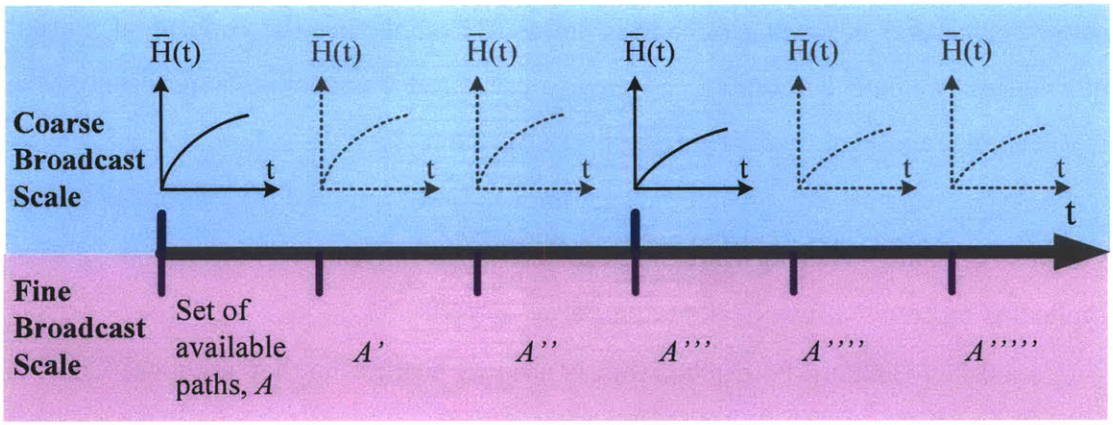

Figure 1-3: A schematic diagram of the broadcast at two time scales. Entropy evolution is broadcast at the coarse time scale; the set of available paths is broadcast at the fine time scale.

signals to the nodes residing on these paths. Available paths will be reserved during the forward part of the probing process. When the destination node is reached, if there is only one path reserved, the destination node sends an ACK to the source node, notifying it with the chosen path. If there are multiple paths reserved, the destination node decides which path to use. It again sends an ACK to notify the source node of the selected path, and also sends release messages along the other reserved paths to release them from the reservations. Once the source node receives the ACK, it starts transmission immediately along the selected path. If no path is reserved, failure of probing is declared. The source node then picks a different set of paths and starts the probing process all over again.

1.3 Key Contributions and Results

Entropy and mutual information are used, for the first time, to quantify the network states and traffic statistics. Entropy is used to quantify the uncertainty of the network states. Higher entropy implicates higher uncertainties. By grouping details of network states into average entropy, we have avoided having to make detailed assumptions about the statistical model of the traffic. Mutual information is used to capture the correlation between neighboring links. Higher mutual information reflects

higher correlation between neighboring links. By employing the concept of mutual information, we have eliminated the need to use detailed probabilistic models to take into account mid-span traffic merging and diverging.

Time evolution of entropy and mutual information estimated from online networks is used to predict the network states in the future. Faster increase of entropy evolution implicates more dynamically changing network traffic.

A modified Bellman-Ford algorithm is used to pick paths that are most likely to be available (lowest entropy) to probe.

Broadcasts at two time scales are used in the control plane as shown in Figure 1-3. Entropy evolution is broadcast with a period of its coherence time (that can range from several minutes to several hours, depending on the actual change in traffic statistics). The set of available paths is broadcast with a period of 0.3 to 0.5 of the average transaction time ($\geq$1S).

Our algorithm is robust against future changes in network topologies and traffic patterns. Instead of using detailed knowledge of network topologies and traffic statistics, we aggregate the information about the internal states of one network domain into one single average entropy, and the correlations between neighboring network domains into mutual information. Therefore, our algorithm is not dependent on detailed assumptions of the models and is much more robust. Furthermore, we have shown that a slight increase in number of paths to probe is well worth the big reduction in network management and control efforts.

Our algorithm reduces control traffic to $1/M$ at the coarse time scale, where M is the number of paths over which we average the evolutions of entropy and mutual information. In [7], detailed traffic statistics of each link in the network is encoded into a control packet, and is broadcast through the control plane transport to each nodes of the network. In contrast, in our algorithm we group the traffic statistics of paths connecting the same pair of nodes into the average evolutions of entropy and mutual information. The average evolutions of entropy and mutual information can be quantized and encoded into a single control packet. By grouping M paths together, we can reduce the control traffic in unit of packets at the coarse time scale

into $\frac{1}{M}$ of that in [7]. In particular, in the wide area network that is going to be discussed in Chapter 4, there can be as many as 100 fiber links connecting one pair of nodes, and for each fiber link there may be about 200 wavelength channels. For such a case, M is about 2000. By grouping the traffic statistics in the 2000 channels into one scalar, the average entropy evolution, we can reduce the amount of control traffic in packets at the coarse time scale by three orders of magnitude.

1.4 Thesis Organization

The rest of the thesis is organized as follows.

In Chapter 2, we analyze the characteristics of the evolution of entropy by studying idealized Markov Process models. We start with the study of a single link. Then we proceed with the study of entropy evolution of a path with L links, assuming independence among the links. Lastly, we discuss how average entropy evolves with time for these two simple cases.

In Chapter 3, we formulate and analyze the problem of determining the number of paths to probe based on the value of entropy for a simple network. First, the problem is formulated based on a simple network model as a Linear Programming problem. Then we obtain a tight upper bound of the expected number of paths to probe given the value of the average entropy. Finally, the tightness of this upper bound is checked against simulation results.

In Chapter 4, we extend the algorithm to a general network which can be a mesh network with multiple-hop paths connecting the nodes. Information Theory techniques enable us to achieve fast-scheduling in a general network without introducing models of traffic statistics while maintaining the capability of including statistical dependencies of the states of neighboring links. Simulation results are provided to verify this algorithm. In addition, a modified Bellman-Ford algorithm is designed to determine the path with the least blocking probability.

In Chapter 5, we discuss the methods to collect in real time the required information: $E[H(t)]$, the expected entropy evolution of each link, and $E[I(t)]$, the expected

time evolution of mutual information of two neighboring links.

Finally, in Chapter 6, we conclude the thesis with a summary of our contributions and discussions of promising future areas of research.

Chapter 2

Evolution of Entropy with Time

In our proposed algorithm, we use the average entropy of links in the set of connections between the same source and destination as a summary of their states. In addition, we use the evolution of the entropy collected from samples of the network states to estimate the link states in the future broadcast intervals. To understand how entropy of the link states evolves with time, in this chapter we start with studying the entropy evolution of a single link using a Markov Process model. We then proceed with the study of entropy evolution of a path with L links, assuming independence among the links. Finally, we discuss how average entropy evolves with time for the two simple cases.

2.1 Entropy Evolution of a Single Link

There are two states for any path: blocked or available. Let $X(t)$ be the probability that the path is blocked at time t. Then we can define the binary entropy of the path at time t to be:

$$H_b(t) = -X(t)\log_2(X(t)) - (1 - X(t))\log_2(1 - X(t)). \tag{2.1}$$

If the path is available at time $t = 0$, $H_b(0)$ equals to zero. As time passes, we become less and less certain about the availability of the path, and $H_b(t)$ increases;

until when we totally lose track of the state of the path, $H_b(t)$ reaches its maximum and can no longer give us any useful information about the path's current state except for its long term average load. Take for example, a path with a Poisson traffic arrival process with arrival rate λ and exponential service distribution time with mean μ as shown in Figure 2-1(a). Assume the state of the link is **0** if it is available to serve traffic, and is **1** if it is otherwise blocked. We can model the link states with a two-state Markov Process, as shown in Figure 2-1(b), with a matrix of transition rate $\mathbf{A} = \begin{bmatrix} -\lambda & \mu \\ \lambda & -\mu \end{bmatrix}$.

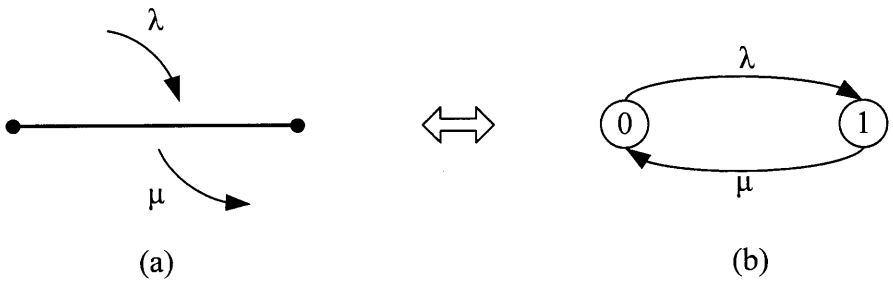

Figure 2-1: The two-state Markov Process model of a single link. (a) A single link with Poisson traffic arrival rate λ and mean service time μ. (b) The Markov Process model of the link states.

The Markov Process is in state **1** if the link is blocked. Therefore, the probability that the Markov Process is in state **1** is $X(t)$, and the probability that the Markov Process is in state **0** is $1 - X(t)$. Let $\mathbf{P(t)} = \begin{bmatrix} 1 - X(t) \\ X(t) \end{bmatrix}$, we can write the Kolmogorov backward equation [13, Chap. 6] of this Markov Process as:

$$\frac{d[\mathbf{P(t)}]}{dt} = \mathbf{A} \times \mathbf{P(t)} \; ; \; t \geq 0. \tag{2.2}$$

With initial condition $\mathbf{P(0)} = [\ 1 \quad 0\ ]^{\mathrm{T}}$, the solution to (2.2) is:

$$\mathbf{P(t)} = e^{-\mathbf{A}t}\mathbf{P(0)} = \begin{bmatrix} \frac{\mu}{\lambda+\mu} + \frac{\lambda}{\lambda+\mu}e^{-(\lambda+\mu)t} \\ \frac{\lambda}{\lambda+\mu} - \frac{\lambda}{\lambda+\mu}e^{-(\lambda+\mu)t} \end{bmatrix}. \tag{2.3}$$

Denote $\rho = \lambda/\mu$ as the loading factor of the link. The entropy of the Markov Process, which is also the entropy of the link states, can be obtained as:

$$\begin{aligned} H(t) = & - X(t)\log_2(X(t)) - (1 - X(t))\log_2(1 - X(t)) \\ = & - (\frac{1}{\rho+1} + \frac{\rho}{\rho+1}e^{-(1+\rho)\mu t}) \times \log_2(\frac{1}{\rho+1} + \frac{\rho}{\rho+1}e^{-(1+\rho)\mu t}) \\ & - (\frac{\rho}{\rho+1} - \frac{\rho}{\rho+1}e^{-(1+\rho)\mu t}) \times \log_2(\frac{\rho}{\rho+1} - \frac{\rho}{\rho+1}e^{-(1+\rho)\mu t}). \end{aligned} \tag{2.4}$$

The entropy evolution of one link is shown in Figure 2-2 and 2-3, in log-log scale and normal scale, respectively. At time zero, we know the Markov Process is in state **0**, and $H(0)$ equals to zero. Observed from both figures, as time passes, we are less confident that the Markov Process is in state **0**, and consequently, $H(t)$ increases. If ρ is smaller than or equal to one, $H(t)$ keeps increasing until it reaches its maximum of one. If ρ is greater than one, $H(t)$ first increases to its maximum of one and then decreases to its steady state value. This corresponds to the cases where links are overloaded. For a heavily loaded link, (e.g., $\rho = 30$ in the two figures), $H(t)$ quickly reaches its maximum and then settles to its steady state value. To see why this happens, we consider the Markov Process in Figure 2-1(b). At time zero, the Markov Process is in state **0**. Since the transition rate from state **0** to state **1** (λ) is much larger than the transition rate from state **1** to state **0** (μ), there is a high probability for the transition from state **0** to state **1** to happen right after the initial state, and $H(t)$ reaches its maximum quickly. As the transition happens quickly, we are overly confident that the Markov Process is in state **1** compared to the confidence level at steady state. On the other hand, if we wait for a longer time, the probability that the Markov Process is in state **1** is determined by the steady state probabilities. Therefore, $H(t)$ decreases to its steady state value once it soars to its maximum.

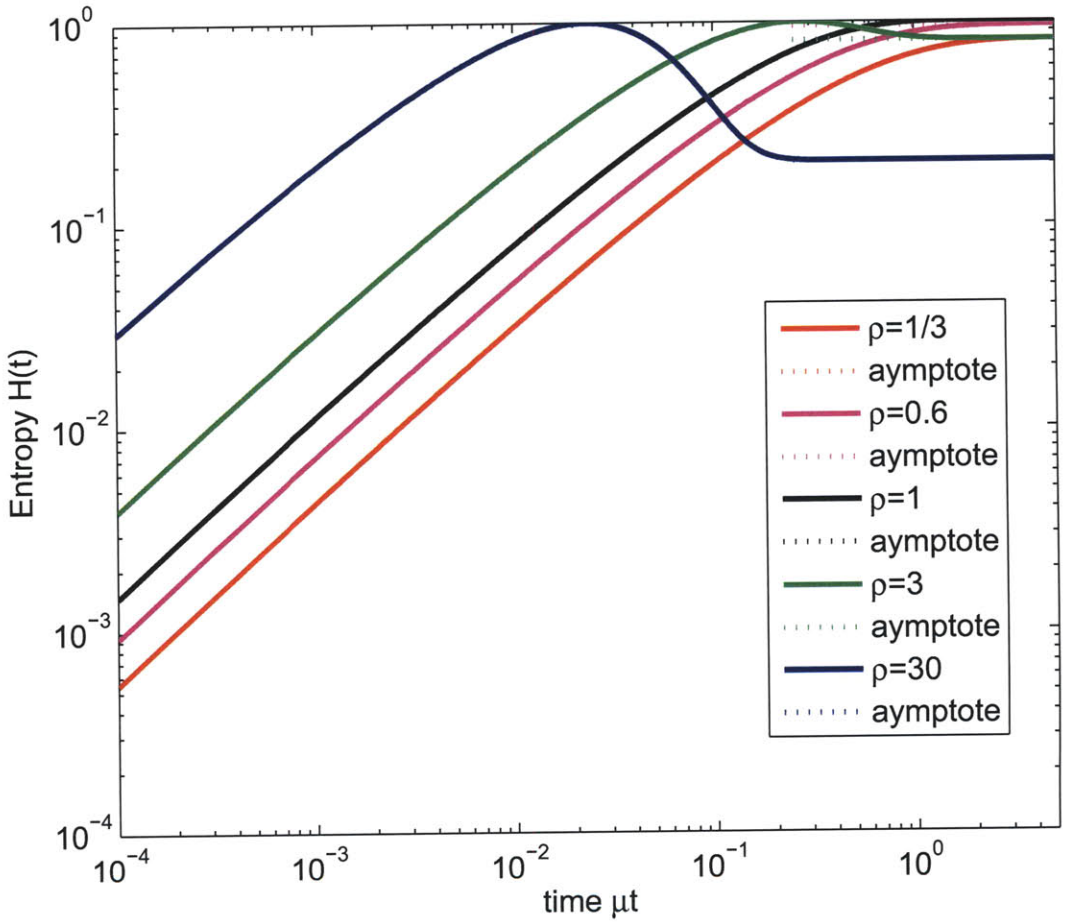

Figure 2-2: Entropy evolution H(t) of one link with loading factor $\rho = 1/3, 0.6, 1, 3$ and 30 in log-log scale. The loading factor ρ is taken as the ratio of the traffic arrival rate λ and the mean service time μ, that is, $\rho = \lambda/\mu$.

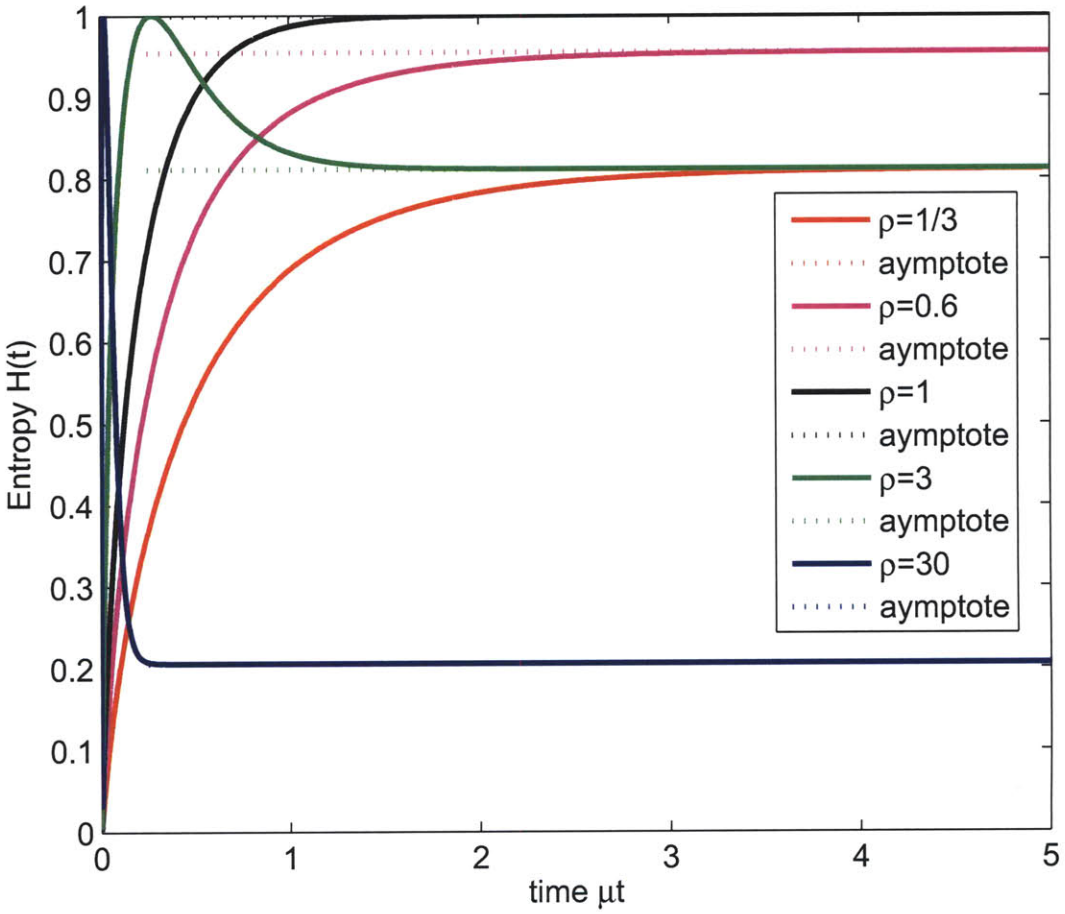

Figure 2-3: Entropy evolution H(t) of one link with loading factor $\rho = 1/3, 0.6, 1, 3$ and 30 in normal scale. The loading factor ρ is taken as the ratio of the traffic arrival rate λ and the mean service time μ, that is, $\rho = \lambda/\mu$.

2.1.1 Entropy at Steady State

By letting t go to infinity, the entropy $H(t)$ of the Markov Process at steady state can be obtained as

$$\begin{aligned} H_\infty &= \lim_{t\to\infty} H(t) \\ &= -\frac{1}{\rho+1}\log_2(\frac{1}{\rho+1}) - \frac{\rho}{\rho+1}\log_2(\frac{\rho}{\rho+1}). \end{aligned} \tag{2.5}$$

H_∞ is only a function of ρ, and is the same if we replace ρ by $\frac{1}{\rho}$. Therefore, if we know whether the link is overloaded or not, we know the loading factor from H_∞.

2.1.2 Time for H(t) to Reach the Maximum

Define t_{max} as the amount of time H(t) takes to increase from zero to its maximum. To get t_{max}, we take the first derivative of $H(t)$ with respect to t:

$$H'(t) = \lambda e^{-(\lambda+\mu)t} \times \log_2(\frac{\mu + \lambda e^{-(\lambda+\mu)t}}{\lambda - \lambda e^{-(\lambda+\mu)t}}). \tag{2.6}$$

By setting $H'(t) = 0$, t_{max} can be calculated as:

$$t_{max} = \begin{cases} \frac{1}{\mu}\cdot\frac{1}{\rho+1}\log(\frac{2\rho}{\rho-1}) & \text{if } \rho > 1 \\ \infty & \text{if } \rho \leq 1 \end{cases}. \tag{2.7}$$

Therefore, for non-overloaded case (i.e., $\rho \leq 1$), $H(t)$ increases to its steady state value at $t = \infty$, while for overloaded case (i.e., $\rho > 1$), $H(t)$ increases to its steady state value at $t = \frac{1}{\mu}\cdot\frac{1}{\rho+1}\log(\frac{2\rho}{\rho-1})$.

Figure 2-4 shows the amount of time for $H(t)$ to increase from zero to its maximum for $\rho > 1$, normalized to the same μ. We can see t_{max} decreases as ρ increases. This is because larger loading factors correspond to larger traffic arrival rates; if the initial state is $\mathbf{0}$, we lose track of the state transitions more quickly for larger arrival rates.

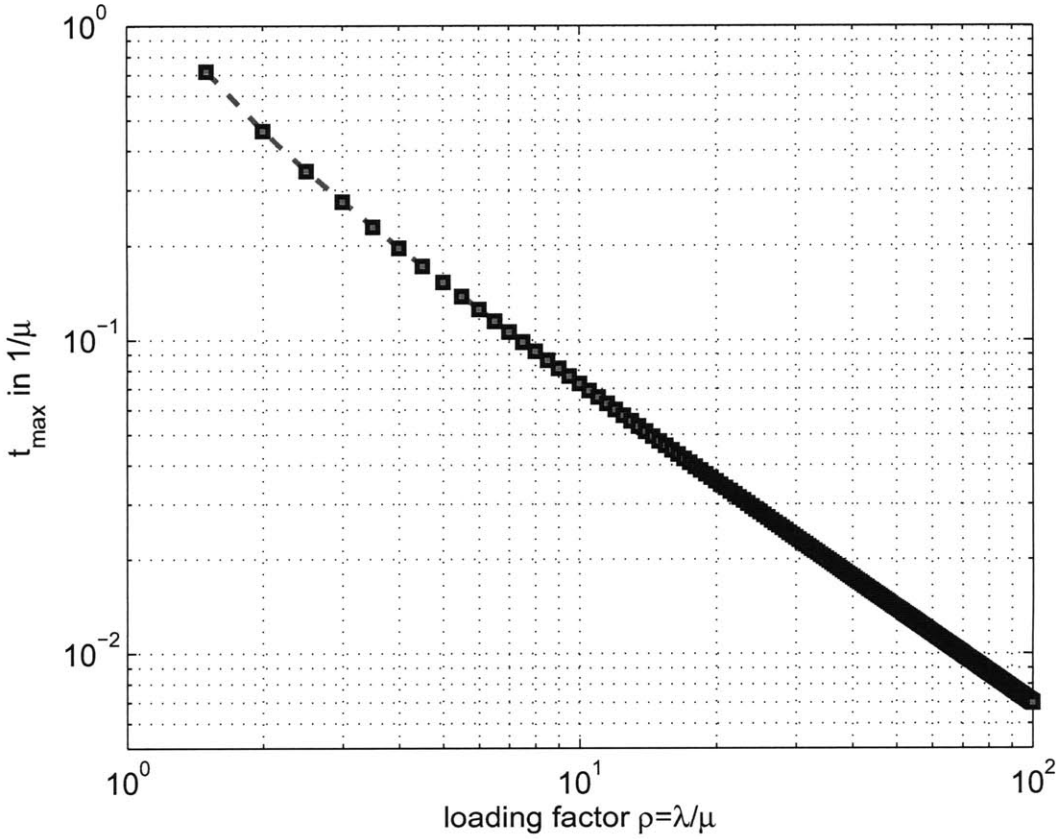

Figure 2-4: The amount of time H(t) takes to reach its maximum, t_{max}, with respect to the loading factor ρ ($\rho = \frac{\lambda}{\mu}$), normalized to the same mean service time μ, in log-log scale.

2.2 Entropy Evolution of a Path with L Independent Links

In real networks, a path is usually composed of several links. The path is available only when all the links are available and is blocked when there is at least one blocked link. To study the entropy evolution of a path with L links (see Figure 2-5), we assume the state of each link is independent of each other, and each of them can be modeled as the same Markov Process in Figure 2-1 (b).

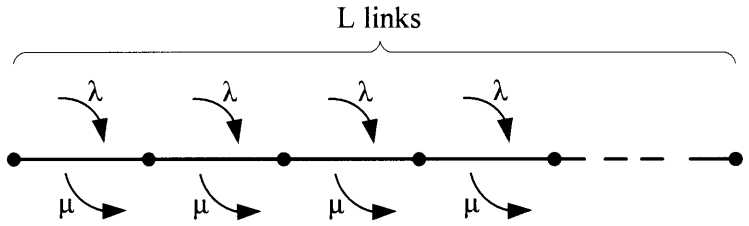

Figure 2-5: A path with L links. The traffic arrival and departure processes of each link are independent Poisson processes with arrival rate λ and departure rate μ.

Similarly, we denote $X_L(t)$ as the blocking probability of the path at time t. If all links are available at time zero, $X_L(t)$ can be calculated as

$$\begin{aligned} X_L(t) &= 1 - (P_0(t))^L \\ &= 1 - (\frac{\mu}{\lambda+\mu} + \frac{\lambda}{\lambda+\mu}e^{-(\lambda+\mu)t})^L. \end{aligned} \tag{2.8}$$

Therefore, applying (2.1), the binary entropy of the state of the path is

$$\begin{aligned} H_L(t) = &- (\frac{1}{\rho+1} + \frac{\rho}{\rho+1}e^{-(1+\rho)\mu t})^L \cdot \log_2[(\frac{1}{\rho+1} + \frac{\rho}{\rho+1}e^{-(1+\rho)\mu t})^L] \\ &- [1 - (\frac{1}{\rho+1} + \frac{\rho}{\rho+1}e^{-(1+\rho)\mu t})^L] \cdot \log_2[1 - (\frac{1}{\rho+1} + \frac{\rho}{\rho+1}e^{-(1+\rho)\mu t})^L]. \end{aligned} \tag{2.9}$$

Figure 2-6 and 2-7 show the entropy evolution for a path with different number of independent links, one in log-log scale and one in normal scale. From both figures, the entropy of a path with a larger loading factor ρ reaches its maximum faster than

that of a path with a smaller ρ. Also, the entropy of a path with a larger L increases faster than that of a path with a smaller L. This can be easily explained as more links brings in more randomness for a path, thus the entropy of the whole path increases faster.

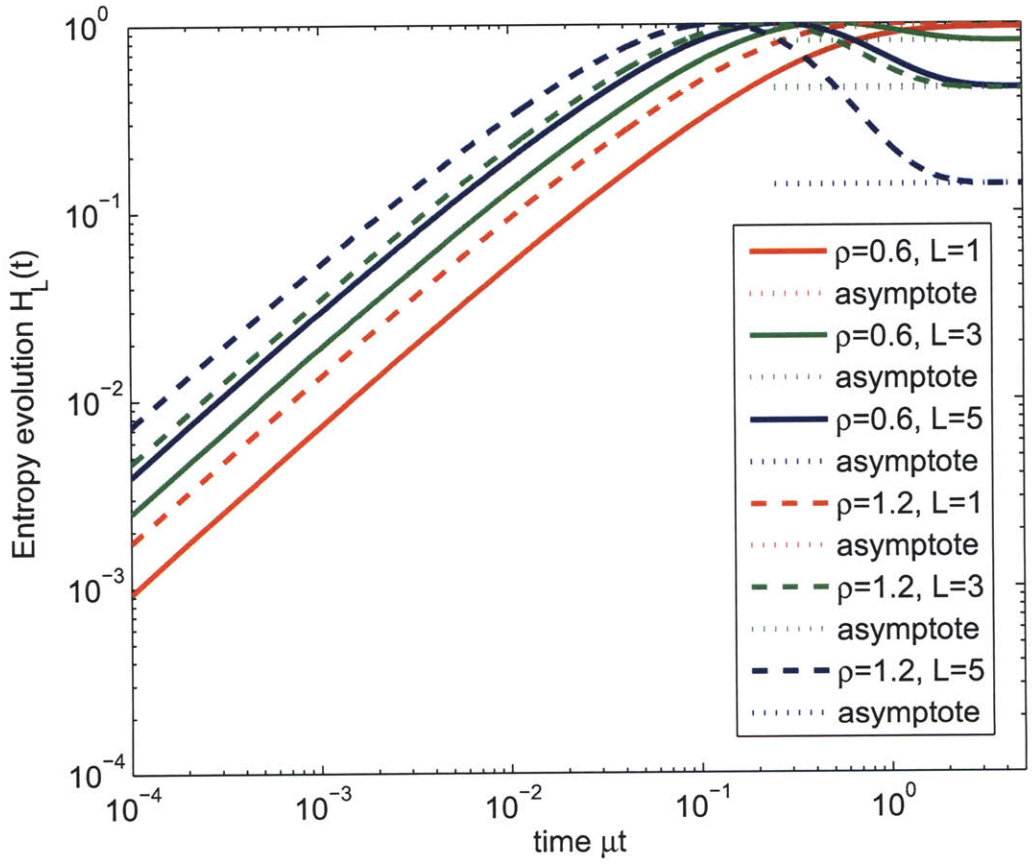

Figure 2-6: Entropy evolution $H_L(t)$ of one path with $L = 1, 3,$ and 5 in log-log scale for $\rho = 0.6$ and 1.2. L is the number of links in a path. ρ is the loading factor for each link.

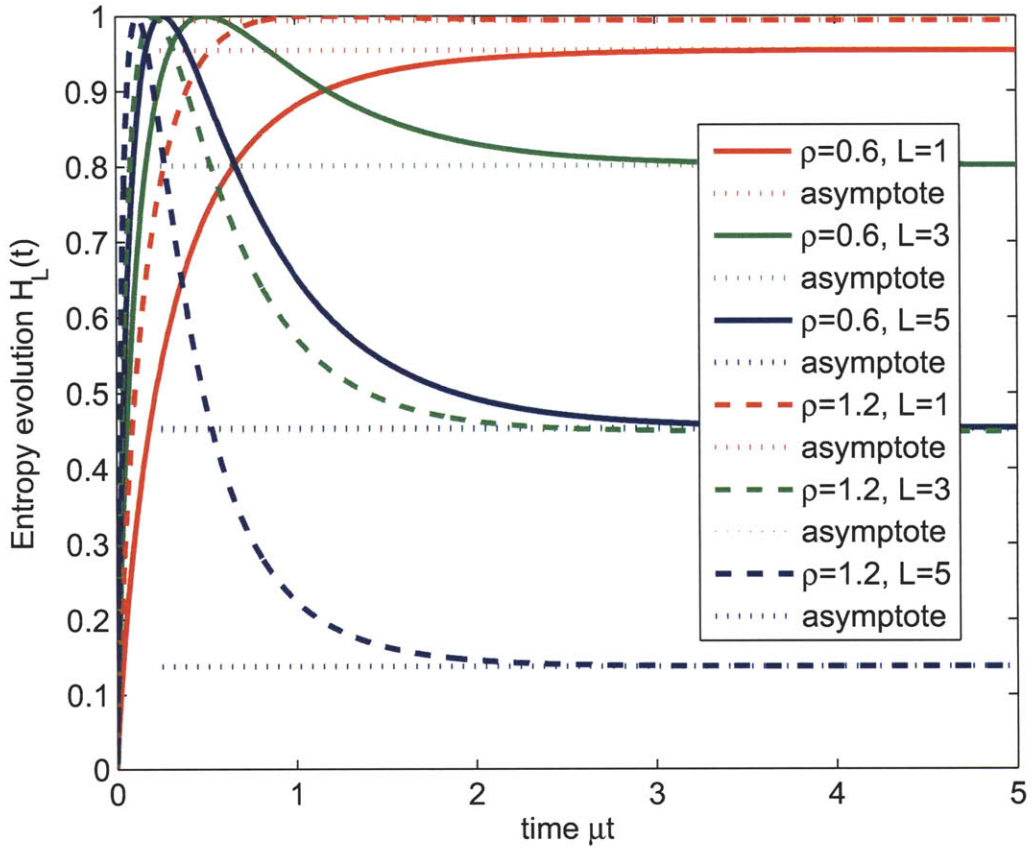

Figure 2-7: Entropy evolution $H_L(t)$ of one path with $L = 1, 3,$ and 5 in normal scale for $\rho = 0.6$ and 1.2. L is the number of links in a path. ρ is the loading factor for each link.

2.2.1 Entropy at Steady State

Letting t go to infinity, the asymptote of $H_L(t)$ can be obtained as:

$$\begin{aligned} H_{L\infty} &= \lim_{t\to\infty} H_L(t) \\ &= -\left(\frac{1}{\rho+1}\right)^L \log_2\left[\left(\frac{1}{\rho+1}\right)^L\right] - \left[1-\left(\frac{1}{\rho+1}\right)^L\right] \log_2\left[1-\left(\frac{1}{\rho+1}\right)^L\right], \end{aligned} \tag{2.10}$$

which is a function of ρ and L.

2.2.2 Time for $H_L(t)$ to Reach the Maximum

Define t_{Lmax} be the amount of time for $H_L(t)$ to increase from zero to its maximum. Similarly as for t_{max}, t_{Lmax} can be obtained as:

$$t_{Lmax} = \begin{cases} \frac{1}{\mu(\rho+1)} \log \frac{\rho}{2^{-\frac{1}{L}}(\rho+1)-1} & \text{if } \rho > 2^{\frac{1}{L}} - 1 \\ \infty & \text{if } \rho \leq 2^{\frac{1}{L}} - 1 \end{cases}, \tag{2.11}$$

For large loading factor ρ, the asymptote of t_{Lmax} is:

$$t_a = \lim_{\rho\to\infty} t_{Lmax} = \frac{1}{(\rho+1)L} \cdot \frac{1}{\mu} \log 2. \tag{2.12}$$

Figure 2-8 depicts t_{Lmax} with respect to ρ for $L = 1, 2$ and 3, normalized to the same μ. t_{Lmax} decreases as ρ and L increases. For large ρ, t_{Lmax} can be approximated by $\frac{1}{(\rho+1)L} \cdot \frac{1}{\mu} \log 2$.

2.3 Summary of Chapter 2

In the previous two sections, we studied entropy evolution of a single link and a single path with L independent links. For both cases, if we are certain about the initial state of the link (or path), as time passes, its entropy either increases to its maximum value and stays there, or first increases to its maximum value and then

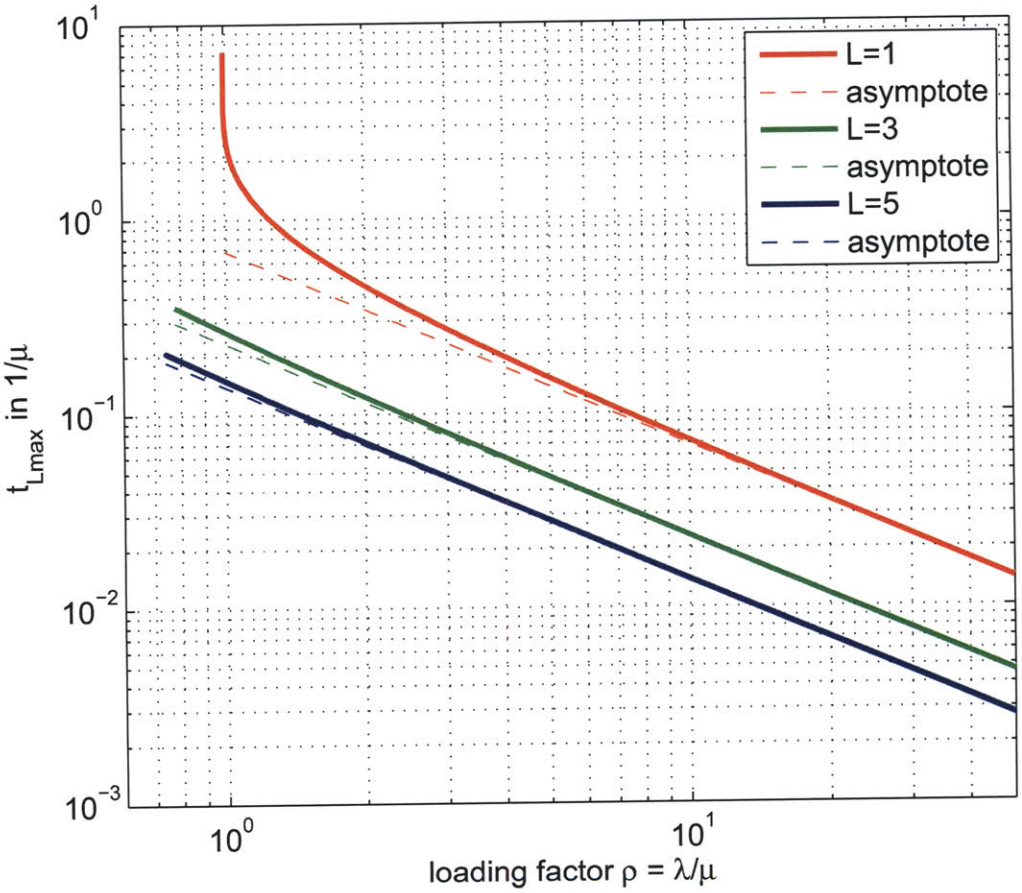

Figure 2-8: The amount of time for $H_L(t)$ to increase from 0 to its maximum, t_{max}, with respect to ρ, normalized to the same μ, in log-log scale. $\rho = \frac{\lambda}{\mu}$ is the loading factor for each constituent link of the path. λ is the traffic arrival rate to each link. μ is the mean service time for transactions at each link.

decreases to its steady state value. With the same traffic statistics for each link, entropy of a path with more than one independent links increases faster than entropy of a single link.

However, in real-life networks, usually traffic statistics of two neighboring links are dependent on each other. If the correlations between any two neighboring links of a path are one, the path can be considered as a single link. If the correlations between any two neighboring links of a path are smaller than one, this can be considered as an intermediate case between the two extremities of a single link and a path with L independent links. Therefore we can argue that the entropy of the path with correlations among its constituent links also increases from zero to its maximum value as time passes.

At the maximum value, entropy loses its ability of inferring the state of the path. Therefore, entropy evolution should be updated before it reaches its maximum. To reduce the complexity for updating and broadcasting entropy evolutions, we do not broadcast $H(t)$ for every path. Instead, we broadcast the set of available links and the average entropy evolution of the group. The average entropy evolution will also starts from zero and keeps increasing within one update broadcast interval.

Chapter 3

Fast Scheduling Algorithm for a Simple Network

Chapter 2 studied how entropy of a path evolves with time. If at the beginning of the fine broadcast interval we know the set of available paths, as time evolves we are less and less certain about the availability of these paths, and the average entropy of the states of these paths increases. Assume for now the evolution of this average entropy can be estimated by online sampling of the networks and is broadcast periodically to each node with a period of the coherence time of the traffic. As discussed in Chapter 1, within one fine broadcast interval, our algorithm picks a number of paths at random from the available set to probe and reserve the paths that are open. When the probing messages reach the destination, if at least one open path has been found, the destination node picks one open path and sends an ACK back to the source, notifying the sender and all the nodes along the chosen reserved light path. It also sends release signals to the nodes along the rest of the reserved paths to release the additional reservations. If no open path is found, the destination node informs the sender on the failure of reservation, and the process is repeated with a new set of paths. On the one hand, we do not want to probe too many paths as we may over reserve resources during probing. On the other hand, we do not want to probe too few paths as we may end up with no open path. Therefore, the problem here is: how to determine the number of paths to be probed to meet a target blocking probability?

In this chapter, we study the problem of determining the number of paths to probe from the entropy evolution for a simple network. We start by introducing a network model in Section 3.1. We then cast the problem as a Linear Programming problem and solve for the upper bound of the expected number of paths to probe. Finally, we verify this upper bound by simulation results.

3.1 Network Model

As discussed in Chapter 1, in our algorithm, we only use the average entropy of a group of paths, instead of the detailed blocking probability of each individual path. Therefore, the paths in one group are considered as having a distribution of blocking probabilities consistent with the reported average entropy. The blocking probability of each path in one group is considered as an independently and identically distributed random variable in our problem formulation.

Figure 3-1 shows the model of a simple network of one source-destination pair with m paths. The blocking probability of each path is an independently and identically distributed random variable X. Set $\mathcal{A}$ is the set of available paths. $N(\mathcal{A})$ is the cardinality (size) of $\mathcal{A}$. Within the $N(\mathcal{A})$ paths, we randomly pick N of them such that their total blocking probability is smaller than or equal to the target blocking probability P_B.

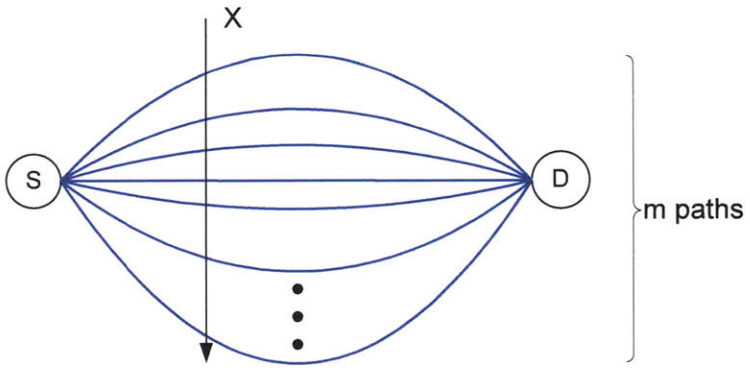

Figure 3-1: A simple network of one source-destination pair with m links in-between. The blocking probability of each link is an independently and identically distributed random variable X.

3.2 Problem Formulation

Mathematically, to meet the target blocking probability P_B, we select N paths out of $\mathcal{A}$ to probe, such that

$$\prod_{i=1}^{N} X_i \leq P_B < \prod_{i=1}^{N-1} X_i,$$

which is equivalent to

$$\sum_{i=1}^{N} \log_2 X_i \leq \log_2 P_B < \sum_{i=1}^{N-1} \log_2 X_i.$$

Taking expectation of both sides,

$$\bar{N} \cdot E[\log_2 X] \leq \log_2 P_B < (\bar{N} - 1) \cdot E[\log_2 X].$$

Therefore, the expected number of path to probe is bounded by

$$\frac{-\log_2 P_B}{E[-\log_2(X)]} + 1 > \bar{N} \geq \frac{-\log_2 P_B}{E[-\log_2(X)]}. \tag{3.1}$$

The average entropy of the network is

$$\bar{H} = \frac{H_1 + H_2 + \cdots + H_{N(\mathcal{A})}}{N(\mathcal{A})}.$$

Taking expectation of both sides, we get

$$E[\bar{H}] = E[\frac{H_1 + H_2 + \cdots + H_{N(\mathcal{A})}}{N(\mathcal{A})}] = E[H].$$

Therefore, for the network model in Figure 3-1, the expected average entropy of all links is equal to the expected entropy of one link. Given the expected average entropy $E[\bar{H}]$, we want to determine the number of paths to probe in terms of the upper and lower bounds of $\bar{N}$. Since, from (3.1), $\frac{-\log_2 P_B}{E[-\log_2(X)]}$ can be less than $\bar{N}$ by at most one, in the following study we approximate $\bar{N}$ by $\frac{-\log_2 P_B}{E[-\log_2(X)]}$.

Let $f_X(x)$ be the density function of the blocking probability X. Since we only

pick paths out of the available set $\mathcal{A}$, and within one fine broadcast interval, the entropy $H(t)$ is increasing, we know the blocking probabilitie of each path in $\mathcal{A}$ is smaller than 0.5. Therefore, we have the following conditions for $f_X(x)$:

$$\begin{aligned} & f_X(x) \geq 0, \text{ for } x \in [0, 0.5], \\ & \int_0^{0.5} f_X(x) d_x = 1, \\ & E[H_b(X)] = h_0, \end{aligned} \tag{3.2}$$

where h_0 is the value of the entropy at time t seconds after the last update.

Let $\mathcal{C}$ be the set of all density functions $f_X(x)$ that satisfy the above conditions. We formulate the following problem to find the lower bound $\bar{N}_{min}$ and upper bound $\bar{N}_{max}$ of $\bar{N}$.

Problem 3.1 *The lower bound of $\bar{N}$ is determined by:*

$$\bar{N}_{min} = \min_{f_X(x) \in \mathcal{C}} \frac{-\log_2 P_B}{E[-\log_2(X)]}. \tag{3.3}$$

Problem 3.2 *The upper bound of $\bar{N}$ is determined by:*

$$\bar{N}_{max} = \max_{f_X(x) \in \mathcal{C}} \frac{-\log_2 P_B}{E[-\log_2(X)]}. \tag{3.4}$$

3.2.1 Lower Bound of $\bar{N}$

To find the lower bound of $\bar{N}$ in Problem 3.1, we need to find $\max_{f_X(x) \in \mathcal{C}} E[-\log_2(X)]$. Clearly, for a density function such as

$$f_X(x) = (1 - h_0)\delta(x) + h_0 \delta(x - 0.5).$$

$f_X(x) \in \mathcal{C}$ and $E[-\log_2(X)] = \infty$. As a result, we have $\bar{N}_{min} = 0$. This is degenerate and a trivial lower bound. We are more interested in $\bar{N}_{max}$ as a working parameter for the algorithm.

3.2.2 Upper Bound of $\bar{N}$

$\bar{N}_{max}$ in Problem 3.2 can be obtained by first solving the following optimization problem.

Problem 3.3 *The minimum of* $E[-\log_2(X)]$ *over* $f_X(x) \in \mathcal{C}$ *is determined by*

$$E[-\log_2(X)]_{min} = \min_{f_X(x)\in\mathcal{C}} E[-\log_2(X)]. \tag{3.5}$$

Problem 3.3 is an optimization problem over the set of continuous functions $f_X(x) \in \mathcal{C}$, which is not easy to solve. But as illustrated in the following discussions, the optimization problem over the discrete counterpart of $f_X(x)$, $P_X(x)$, is a Linear Programming problem. More importantly, as will be discussed later, the minimum of $E[-\log_2(X)]$ we get from the Linear Programming problem is also optimum for the optimization over the continuous case of $f_X(x)$.

To optimize (3.5) over $P_X(x)$, let $x_1, x_2, ..., x_n$ be any n different possible values for X in $[0, 0.5]$, and $y_1, y_2, ..., y_n$ be the probability weights for $x_1, x_2, ..., x_n$ (i.e., $P_X(x_i) = y_i$). Then the conditions in (3.2) can be rewritten as

$$\begin{aligned} &\sum_{i=1}^{n} y_i = 1, \\ &\sum_{i=1}^{n} y_i H_b(x_i) = h_0, \\ &y_i \geq 0, \text{ for } i \in \{1, ..., n\}. \end{aligned} \tag{3.6}$$

Subjecting to the above conditions, we want to minimize $\sum_{i=1}^{n} y_i[-log_2(x_i)]$.

To further transform the conditions and the problem, we define the following vectors:

- $\mathbf{1} = [\ 1 \quad 1 \quad \ldots \quad 1\ ]^{\mathrm{T}}$
- $\mathbf{y} = [\ y_1 \quad y_2 \quad \ldots \quad y_n\ ]^{\mathrm{T}} \geq 0$
- $\mathbf{h} = [\ H_b(x_1) \quad H_b(x_2) \quad \ldots \quad H_b(x_n)\ ]^{\mathrm{T}} \geq 0$

- $\mathbf{g} = [\ -\log_2(x_1) \quad -\log_2(x_2) \quad \ldots \quad -\log_2(x_n)\ ]^{\mathrm{T}} \geq 0$

Now the conditions in (3.6) are equivalent to:

$$\begin{aligned} &\mathbf{1}^{\mathrm{T}}\mathbf{y} = 1, \\ &\mathbf{h}^{\mathrm{T}}\mathbf{y} = h_0, \\ &y_i \geq 0, \text{ for } i \in \{1, ..., n\}. \end{aligned} \tag{3.7}$$

Let $\mathcal{Y}$ be the polyhedron defined by $\mathbf{y}$ under the conditions in (3.7). With the new representations, the discrete case of Problem 3.3 is:

Problem 3.4 *The minimum of* $E[-\log_2(X)]$ *over the discrete case of* $f_X(x)$ *(i.e. probability weights represented by* $\mathbf{y} \in \mathcal{Y}$*), is determined by*

$$E[-\log_2(X)]_{min} = \min_{\mathbf{y}\in\mathcal{Y}} \mathbf{g}^{\mathrm{T}}\mathbf{y}. \tag{3.8}$$

Problem 3.4 is an Linear Programming problem. For the Linear Programming problem of minimizing $\mathbf{g}^T\mathbf{y}$ over $\mathbf{y} \in \mathcal{Y}$, if there exists an optimal solution, there exists a basic feasible optimal solution [3, Chap. 2], denoted by $\mathbf{y}^*$. For a basic feasible solution, there are n linearly independent active constraints on $\mathbf{y}^*$. In conditions (3.7), we already have two such constraints, $\mathbf{1}^{\mathrm{T}}\mathbf{y} = 1$ and $\mathbf{h}^{\mathrm{T}}\mathbf{y} = h_0$. Therefore, we need $(n-2)$ y_i's such that $y_i = 0$. Intuitively, since $\mathbf{g} \geq 0$ and $\mathbf{y} \geq 0$, the minimum of $\mathbf{g}^T\mathbf{y}$ is achieved by letting the $(n-2)$ y_i's for the largest $(n-2)$ g_i's equal to zero.

As a consequence, for any chosen set of discrete values of X, the optimization problem 3.4 can always reduces to a problem where only two of the y_i^*'s are greater than or equal to zero. What's more, the optimum result obtained by having only two of the y_i^*'s are greater than or equal to zero in Problem 3.4 is also optimum for Problem 3.3, as illustrated by the following theorem.

Theorem 3.1 *Let* X_c *be a continuous random variable with probability density function* $f_{X_c}(x) \in \mathcal{C}$*. Let* X_d^* *be the discrete random variable that achieves optimum solution in Problem 3.4. Then,* $E[-\log_2(X_c)] \geq E[-\log_2(X_d^*)]$.

The proof of the above theorem is shown in A.1. Therefore, one $f_X(x)$ that optimizes Problem 3.3 can be written as:

$$f_X(x) = \alpha\delta(x - x_1) + (1 - \alpha)\delta(x - x_2),$$

where $\alpha \in (0, 1)$, $x_1 \in (0, 0.5)$, and $x_2 \in (0, 0.5)$. As a result, Problem 3.3 can be transformed to the following problem.

Problem 3.5 *The minimum of* $E[-\log_2(X)]$ *over* $f_X(x) \in \mathcal{C}$ *is determined by*

$$\begin{aligned} E[-\log_2(X)]_{min} = \min_{\alpha\in[0,1],x_1,x_2\in[0,0.5]} & -\alpha\log_2(x_1) - (1-\alpha)\log_2(x_2) \\ \textit{subject to:} & -\alpha H_b(x_1) - (1-\alpha)H_b(x_2) = h_0. \end{aligned} \tag{3.9}$$

With this transformation, the optimal solution to Problem 3.3 can be readily solved as (see A.2):

$$E[-\log_2(X)]_{min} = \begin{cases} -\log_2[H_b^{-1}(h_0)] & \text{if } h_0 \le h_A \\ \frac{(1-h_0)\{-\log_2[H_b^{-1}(h_0)]\}+h_0-h_A}{1-h_A} & \text{if } h_0 > h_A \end{cases}, \tag{3.10}$$

where $H_b^{-1}(h_0)$ is the inverse function of $H_b(x) = h_0$ for $x \in (0, 0.5)$. h_A is the solution to $\frac{h-1}{H_b^{-1}(h)\log 2\cdot\log_2\frac{1-H_b^{-1}(h)}{H_b^{-1}(h)}} - log_2H_b^{-1}(h) - 1 = 0$ and, numerically, $h_A \approx 0.4967$.

Figure 3-2 depicts $E[-\log_2(X)]_{min}$ in (3.10) for $P_B = 10^{-4}$. As discussed earlier, $E[-\log_2(X)]_{min}$ equals to $-\log_2[H_b^{-1}(h_0)]$ for h_0 smaller than or equal to h_A, and is smaller for h_0 greater than h_A. However, from the figure, we can see $-\log_2[H_b^{-1}(h_0)]$ is a close approximation for $E[-\log_2(X)]_{min}$ even when h_0 is greater than h_A. We will discuss more on the justification of this approximation of $E[-\log_2(X)]_{min}$ using $-\log_2[H_b^{-1}(h_0)]$ later.

Now substituting (3.10) into (3.4), we obtain $\bar{N}_{max}$ in Problem 3.2 as:

$$\bar{N}_{max} = \begin{cases} \frac{-\log_2(P_B)}{-\log_2[H_b^{-1}(h_0)]} & \text{if } h_0 \le h_A \\ \frac{-\log_2(P_B)\cdot(1-h_A)}{(1-h_0)\{-\log_2[H_b^{-1}(h_0)]\}+h_0-h_A} & \text{if } h_0 > h_A \end{cases}. \tag{3.11}$$

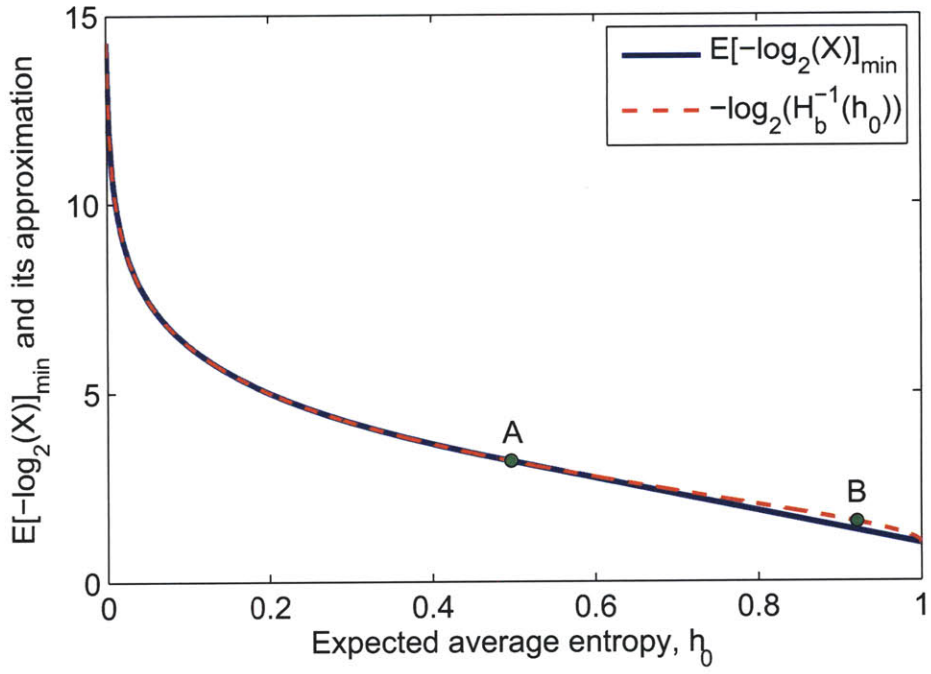

Figure 3-2: $E[-\log_2(X)]_{min}$ and its approximation for $P_B = 10^{-4}$. P_B is the target blocking probability.

Similar to the approximation of $E[-\log_2(X)]_{min}$ using $-\log_2[H_b^{-1}(h_0)]$, $\bar{N}_{max}$ can be approximated by $\bar{N}_{app}$, which is defined as

$$\bar{N}_{app} = \frac{-\log_2(P_B)}{-\log_2[H_b^{-1}(h_0)]}. \tag{3.12}$$

Figure 3-3 plots $\bar{N}_{max}$ and $\bar{N}_{app}$ with respect to the expected average entropy. $\bar{N}_{app}$ is the same as $\bar{N}_{max}$ for $h_0 \leq h_A$ and is smaller than $\bar{N}_{max}$ for $h_0 > h_A$. Both $\bar{N}_{app}$ and $\bar{N}_{max}$ increase as h_0 increases. For h_0 smaller than 0.1, we know the paths in $\mathcal{A}$ have low blocking probabilities. Therefore the average number of paths we need to probe is only one or two, i.e. $\bar{N}_{max} < 2$. For h_0 close to 1, we are less certain about the availability of the paths in $\mathcal{A}$. Thus, we end up with probing more of them. The largest difference between $\bar{N}_{max}$ and $\bar{N}_{app}$ occurs at point B in Figure 3-3, where $\bar{N}_{app}$ is smaller than $\bar{N}_{max}$ by:

$$\frac{\bar{N}_{max} - \bar{N}_{app}}{\bar{N}_{max}}|_{h_B} = 0.145.$$

This leads to a difference of only one or two paths for $P_B = 10^{-4}$, for which case $\bar{N}_{app}$ can be taken as a good approximation of $\bar{N}_{max}$. In fact for the entropy technique to be useful, most of the time the network will be operating with entropy less than h_A, where the two expressions are equal.

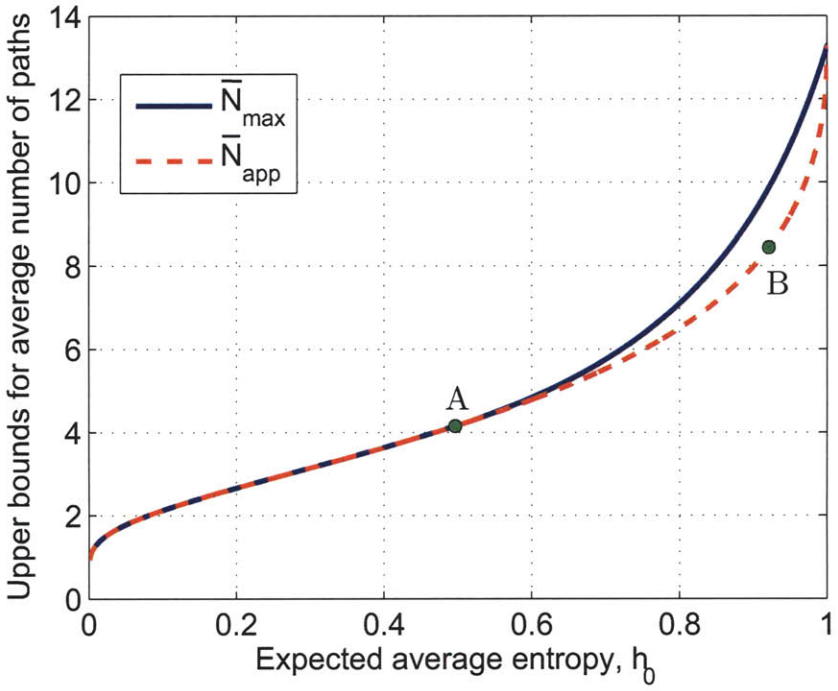

Figure 3-3: $\bar{N}_{max}$ and $\bar{N}_{app}$ for $P_B = 10^{-4}$. $\bar{N}_{max}$, defined in (3.11), is the maximum of the expected number of paths to probe given the expected average entropy is h_0. $\bar{N}_{app}$, defined in (3.12), is the approximation of $\bar{N}_{max}$. $P_B = 10^{-4}$ is the target blocking probability.

The number of paths to probe is an integer. This can be obtained by rounding up $\bar{N}_{max}$ to the integer ceiling of $\bar{N}_{max}$. Figure 3-4 shows the ceilings of $\bar{N}_{max}$ and $\bar{N}_{app}$ for $P_B = 10^{-4}$.

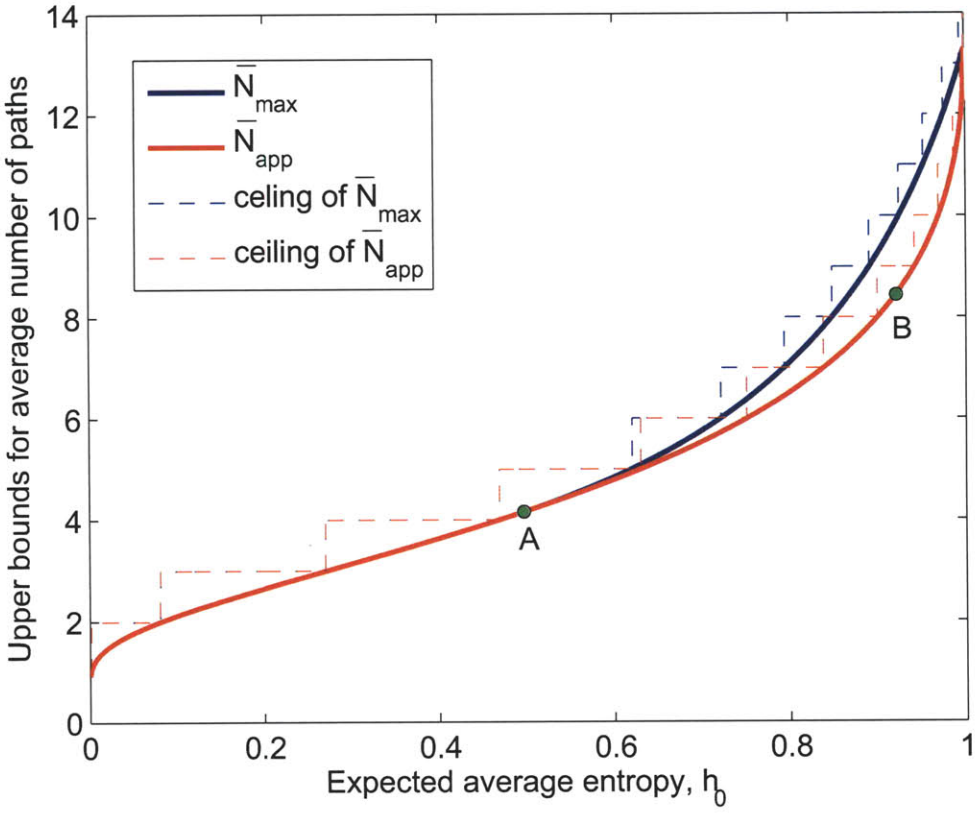

Figure 3-4: $\bar{N}_{max}$ and $\bar{N}_{app}$ and their ceilings for $P_B = 10^{-4}$. Ceiling of $\bar{N}_{max}$ is taken as the integer ceiling of $\bar{N}_{max}$. Similarly, ceiling of $\bar{N}_{app}$ is the integer ceiling of $\bar{N}_{app}$.

3.3 Simulation Results and Theoretical Bounds

To evaluate the performance of the proposed method of determining the number of paths to probe based on the expected average entropy value, computer simulation results are presented in this section. The simulation was based on the model in Figure 3-1. The basic idea is to simulate a simple network of one source-destination pair with m links in between. In our algorithm, we only use the average entropy to determine the average number of paths to probe. In order to compare the performance of our algorithm with the actual average number of paths to probe, each link in the simulation is assigned with a randomly drawn blocking probability according to the same probability density function. Based on these randomly assigned blocking

probabilities, we can calculate the average entropy and thus the theoretical bound of the expected number of paths to probe. We can also find the actual average number of paths to probe if the paths are randomly chosen in the simulation. Therefore, the comparison can be carried out.

Two density functions were used to generate blocking probabilities, uniform and truncated Gaussian. Section 3.3.1 discusses the simulation results with blocking probabilities drawn from uniform distributions, while Section 3.3.2 discusses the simulation results with blocking probabilities drawn from truncated Gaussian distributions.

3.3.1 Uniform Distribution of Blocking Probability X

Based on the model in Figure 3-1, we simulated a simple network of one source-destination pair with m links in between. A randomly drawn blocking probability with uniform distribution in $[0, 0.5]$ is assigned to each path. As shown in Figure 3-5, two forms of N as functions of the average entropy value h_0, N_r and N_o, are plotted. To get N_r, a sequence of paths are randomly selected from the pool of m available paths until the total blocking probability of the selected paths is smaller than the target blocking probability P_B. N_r is taken as the average of the numbers of paths of such repeated processes. On the other hand, to get N_o, paths are picked in ascending order of their blocking probabilities, that is, the path with lowest blocking probability is picked first, followed by the one with the second lowest blocking probability, etc., until their total blocking probability is smaller than P_B. Then, in the same manner as for N_r, N_o is taken as the average number of selected paths over many runs. In particular, N_o can be considered as the analogue of the case from [7] for heterogeneous traffic arrival and departure processes. Since N_o assumes knowing the individual blocking probabilities of each path and N_r does not, our algorithm will choose $\bar{N}_{max}$ paths to probe, which is very close to N_r.

Figure 3-5 shows the simulation results of N_r and N_o in comparison with $\bar{N}_{max}$, $\bar{N}_{max}+1$ and $\bar{N}_{app}$. Figure 3-6 shows the numbers of paths and their ceilings. In both figures, we observe that $\bar{N}_{max}+1$ is the upper bound for N_r, and $\bar{N}_{max}$ and $\bar{N}_{app}$ are close approximations of N_r. In Figure 3-5, N_r is bounded by $\bar{N}_{max}$ for $h > 0.83$, and

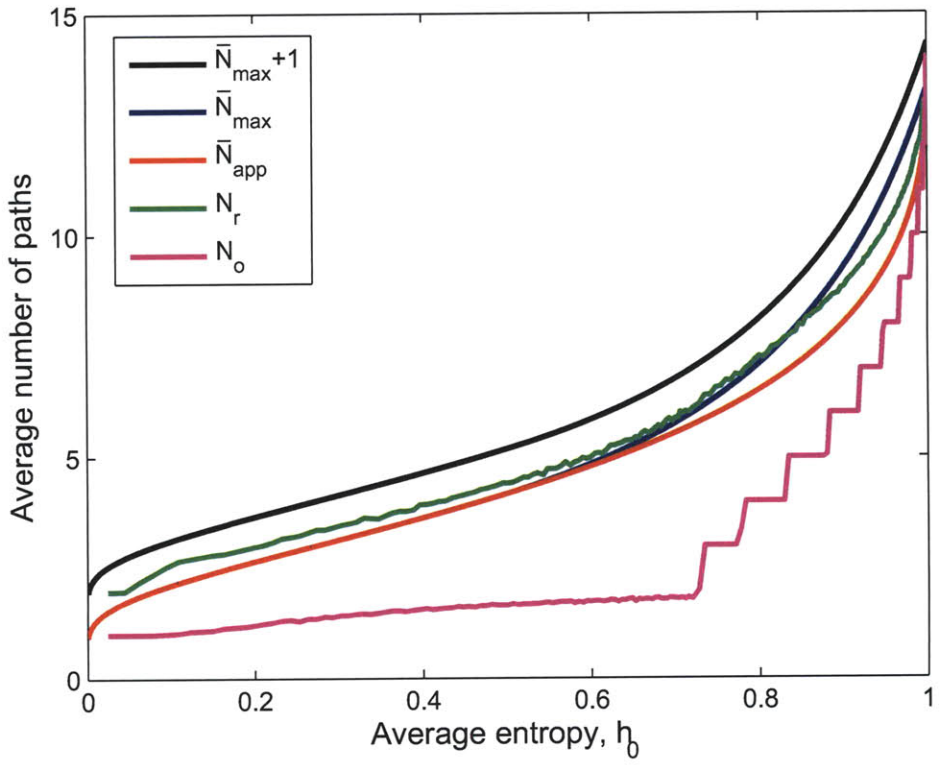

Figure 3-5: Simulation results of average number of paths to probe given $E[\bar{H}] = h_0$ to achieve $P_B = 10^{-4}$, for the case of uniformly-distributed blocking probability for each link. $\bar{N}_{max}$, defined in (3.11), is the theoretical upper bound of the expected number of paths to probe. $\bar{N}_{app}$,defined in (3.12), is the approximation of $\bar{N}_{max}$. N_r is the simulated average number of paths to probe when only the average entropy is given and paths are selected randomly. N_o is the simulated average number of paths to probe when detailed blocking probability of each path is given. $P_B = 10^{-4}$ is the target blocking probability.

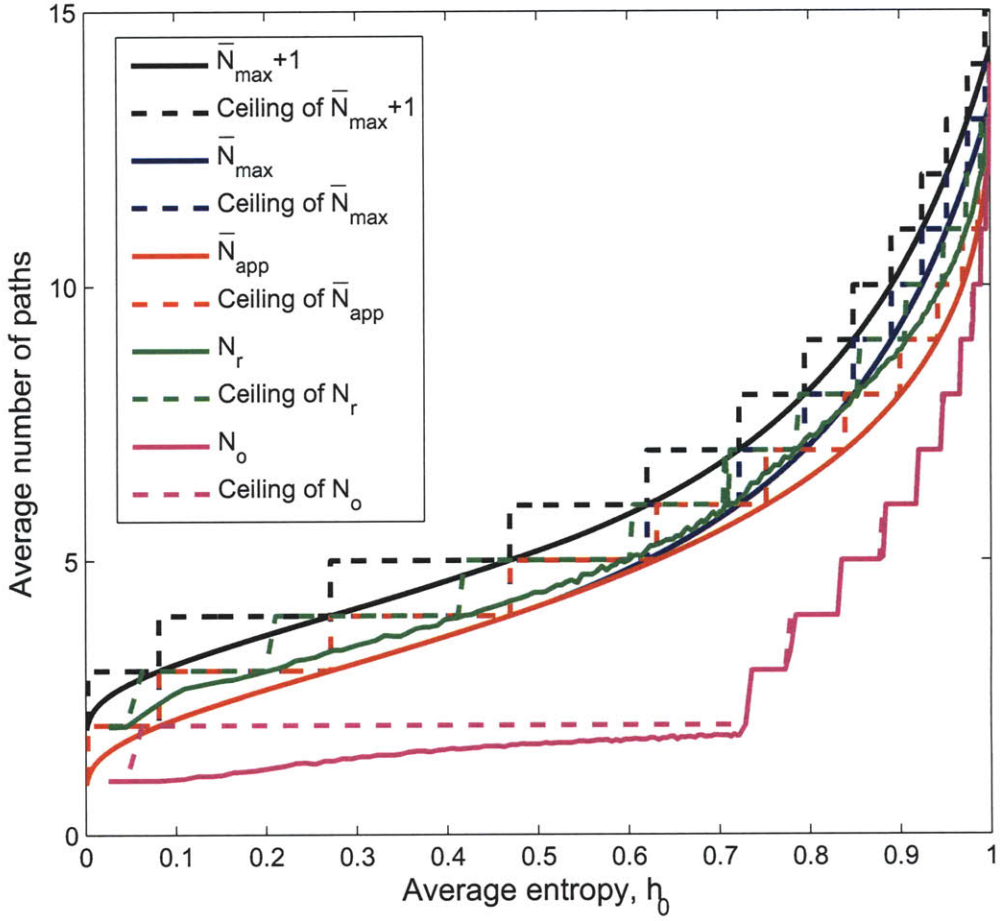

Figure 3-6: Simulation results of average number of paths to probe with ceilings given $E[\bar{H}] = h_0$ to achieve $P_B = 10^{-4}$, for the case of uniformly-distributed blocking probability for each link. $\bar{N}_{max}$ is defined in (3.11). $\bar{N}_{app}$ is defined in (3.12). N_r is the simulated average number of paths to probe when only the average entropy is given and paths are selected randomly. N_o is the simulated average number of paths to probe when detailed blocking probability of each path is given. $P_B = 10^{-4}$ is the target blocking probability. Ceiling of $\bar{N}_{max}$ is the integer ceiling of $\bar{N}_{max}$. Ceilings of other lines are obtained similarly.

is slightly larger than $\bar{N}_{max}$ for $h < 0.83$. However, the latter case can be justified by the approximation applied in the problem formulation in Section 3.2, where $\bar{N}_{max}$ is actually confined by (3.1). Indeed, observing Figure 3-5, even when N_r is larger than $\bar{N}_{max}$, N_r is always smaller than $\bar{N}_{max} + 1$. Therefore, $\bar{N}_{max} + 1$ is a good upper bound for N_r and $\bar{N}_{max}$ is very close to N_r. In addition, $\bar{N}_{app}$ is no smaller than N_r by one for all h values, which suggests it is a good approximation to N_r as well. On the other hand, N_o is smaller than N_r for all $h \in (0, 1)$, and is only half of N_r for $h \in (0.4, 0.78)$. Nevertheless, this is understandable as we have to sacrifice some performance in order to avoid detailed assumptions of network statistics and to reduce the amount of network control and management messaging.

Figure 3-7 and 3-8 show the results of simulations for different target blocking probabilities, $P_B = 10^{-2}$, 10^{-4}, and 10^{-6}. The parameters defined are the same as those in Figure 3-5 and 3-6. From the figures we can see the average number of paths to probe increases linearly with $-\log_2 P_B$. For smaller P_B, N_r is bounded by $\bar{N}_{max}$ in a wider range of entropy h_0, while N_r is always bounded by $\bar{N}_{max} + 1$. This is due to the rounding-up effect incurred when the actual blocking probability of $N + 1$ paths is smaller than P_B but the total blocking probability of N paths is greater than P_B. When P_B is larger, it is easier to overshoot the blocking probability requirement with fewer number of paths. Therefore, the rounding-up effect is larger for larger P_B. However, N_r is bounded by $\bar{N}_{max} + 1$ for all values of P_B, which can be clearly observed from the figures.

3.3.2 Truncated Gaussian Distribution of Blocking Probability X

Similarly to the case of uniformly distributed blocking probability for each link, we simulated the simple network in Figure 3-1 with each link assigned a randomly drawn blocking probability with truncated Gaussian distributions. The Gaussian distributions used in the simulation are truncated to be within interval $[0, 0.5]$, with means varying from zero to 0.5 and standard deviation 0.1. This ensures us to get a wide

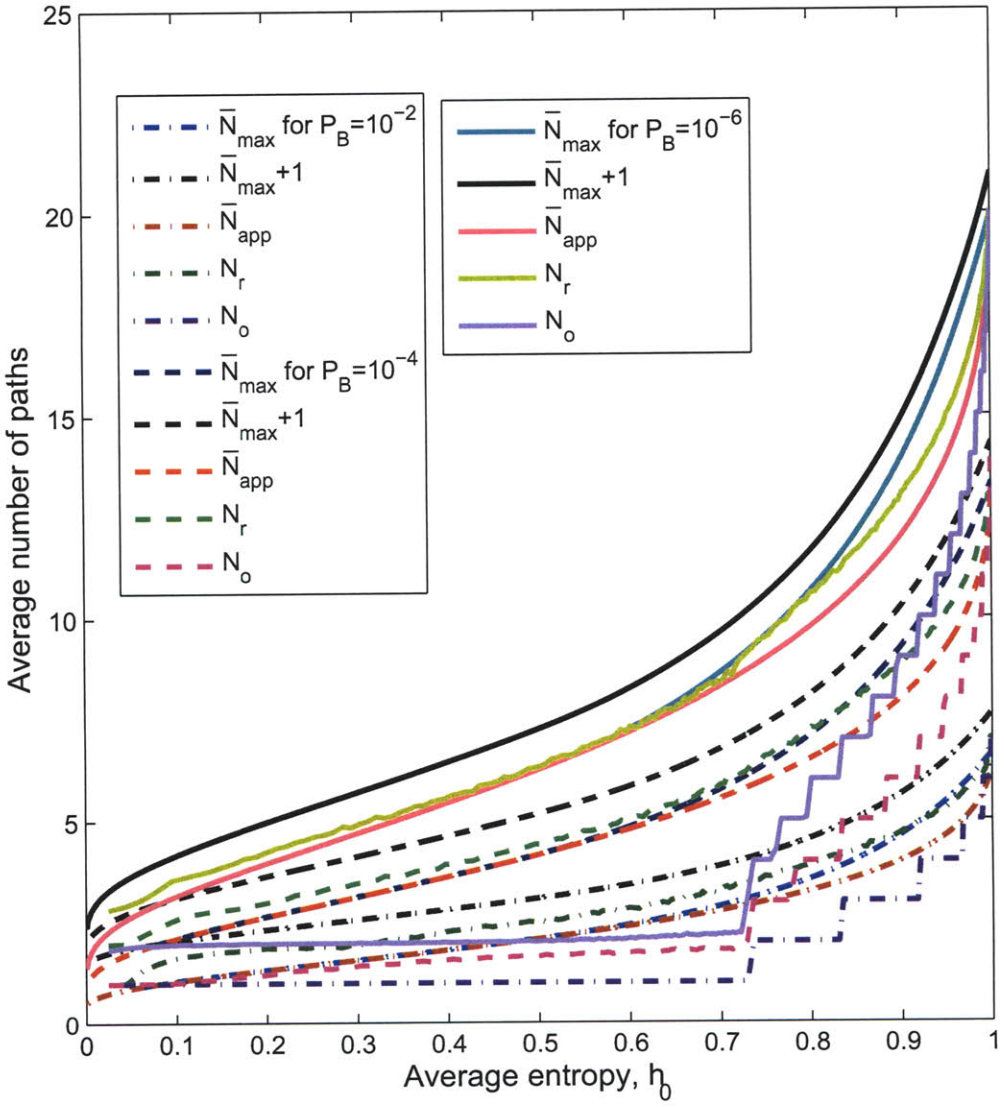

Figure 3-7: Average number of paths to probe given $E[\bar{H}] = h_0$ to achieve the target blocking probabilities $P_B = 10^{-2}$, 10^{-4}, and 10^{-6} in the simulation of uniformly-distributed blocking probability for each link. N_r is the simulated average number of paths to probe when only the average entropy is given and paths are selected randomly. N_o is the simulated average number of paths to probe when detailed blocking probability of each path is given. P_B is the target blocking probability.

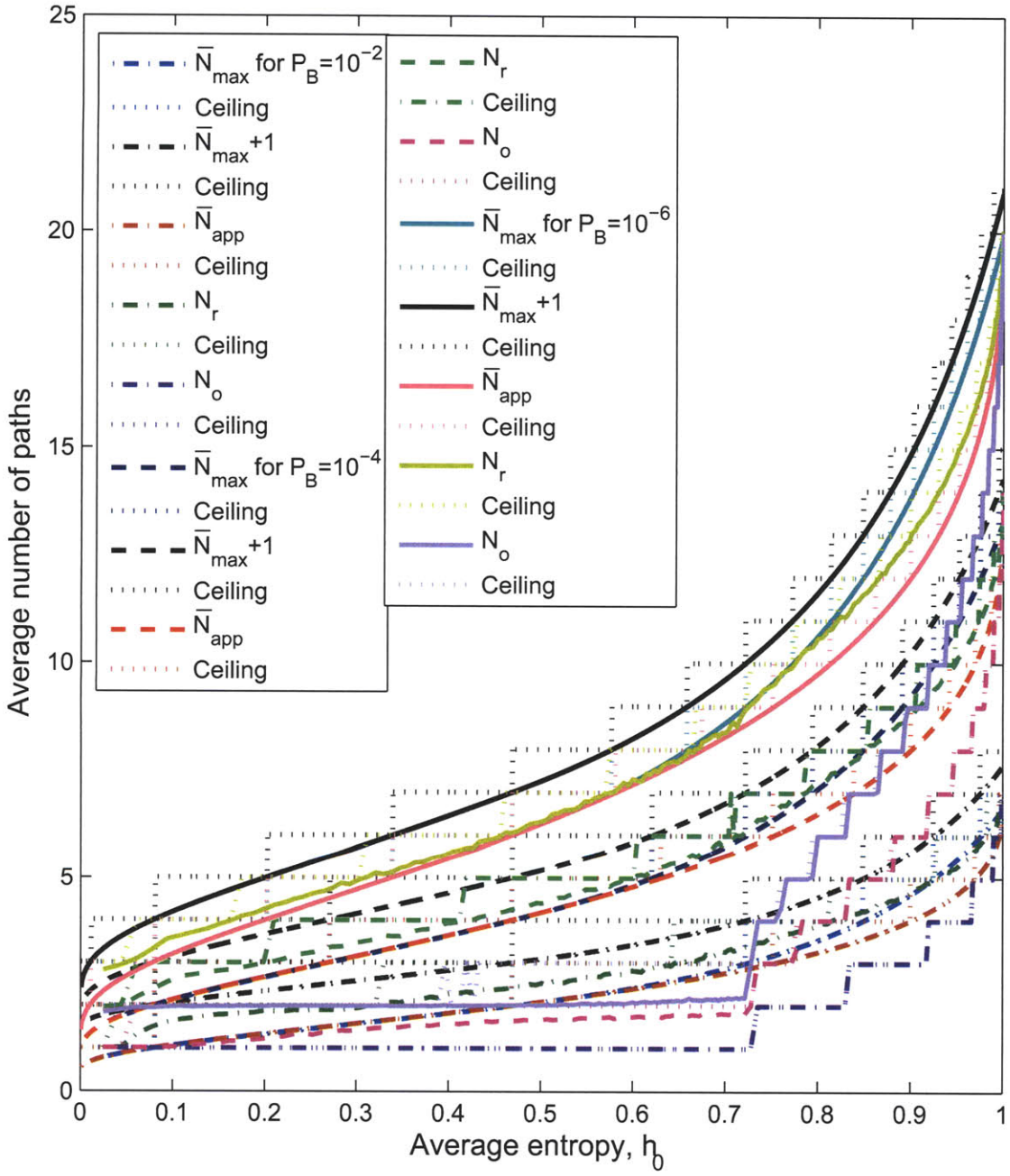

Figure 3-8: Average number of paths to probe with ceilings given $E[\bar{H}] = h_0$ to achieve $P_B = 10^{-2}$, 10^{-4}, and 10^{-6} in the simulation of uniformly-distributed blocking probability for each link.

range of average entropy value. Figure 3-9 and 3-10 show two forms of N, N_r and N_o, which are obtained in similar ways as discussed in the previous section. Figure 3-10 shows the same simulation results as Figure 3-9 except with ceilings of the numbers of paths.

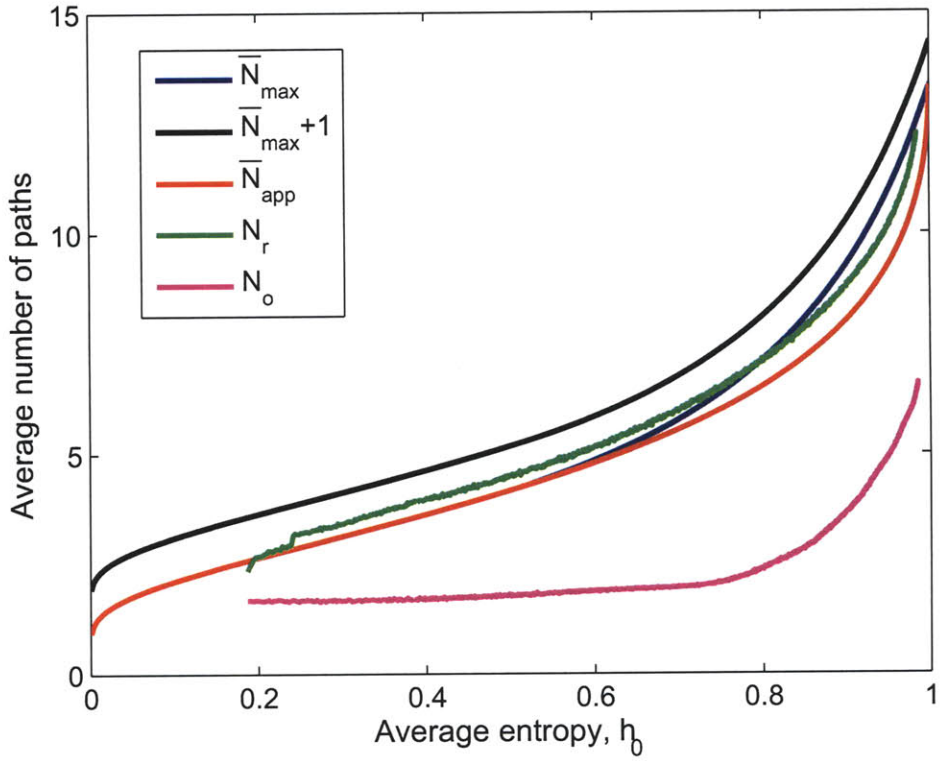

Figure 3-9: Simulation results of average number of paths to probe given $E[\bar{H}] = h_0$ to achieve $P_B = 10^{-4}$, for the case of truncated Gaussian-distributed blocking probability for each link. $\bar{N}_{max}$, defined in (3.11), is the upper bound of the expected number of paths to probe given the expected average entropy. $\bar{N}_{app}$, defined in (3.12), is the approximation of $\bar{N}_{max}$. N_r is the simulated average number of paths to probe when only the average entropy is given and paths are selected randomly. N_o is the simulated average number of paths to probe when detailed blocking probability of each path is given. P_B is the target blocking probability.

From Figure 3-9 and 3-10, we also observe that $\bar{N}_{max} + 1$ is a good upper bound for N_r; $\bar{N}_{max}$ and $\bar{N}_{app}$ are close approximations to N_r. With detailed information, we only need to pick on average N_o number of paths, which is smaller than N_r as expected.

Figure 3-11 and 3-12 show the results of simulations for different target blocking

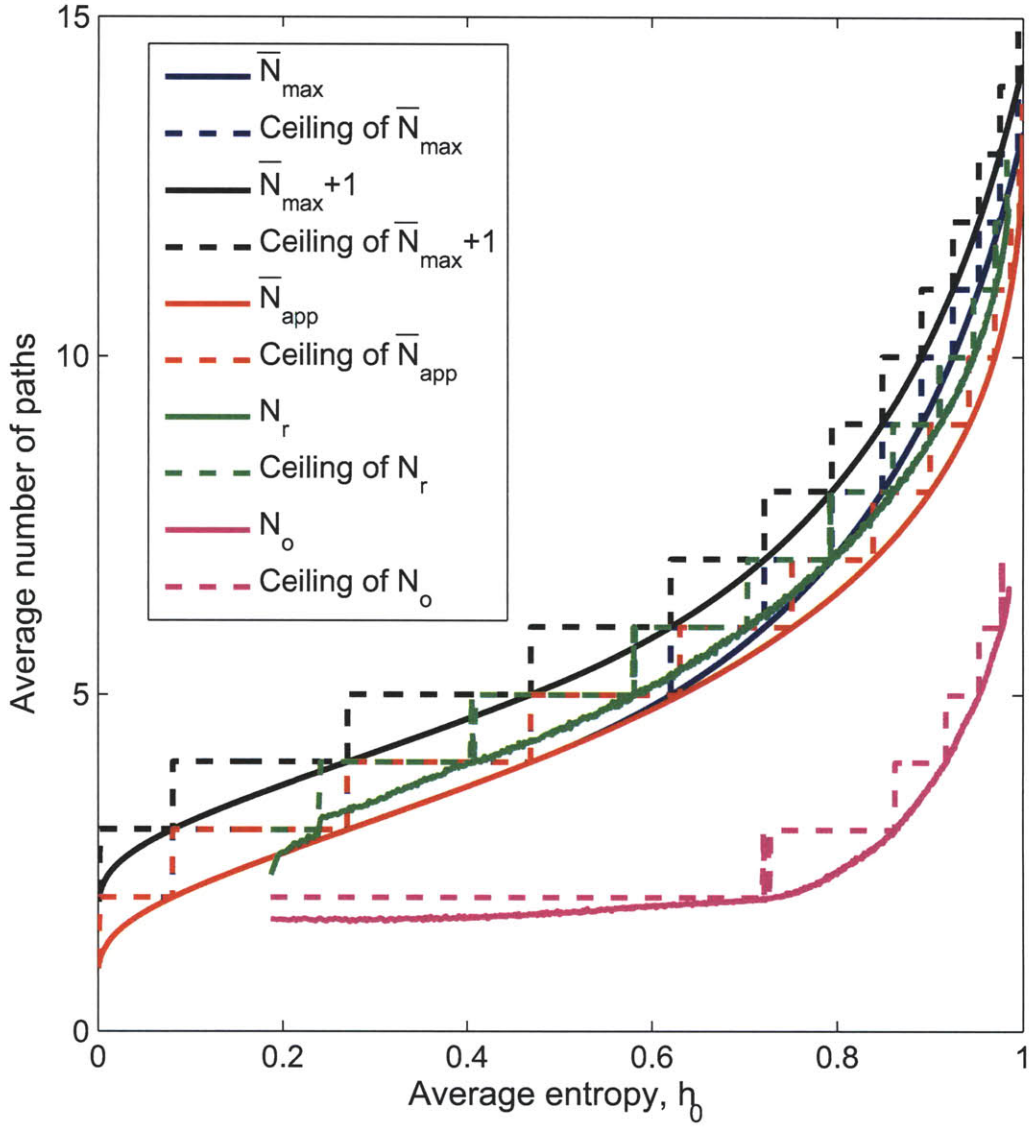

Figure 3-10: Simulation results of average number of paths to probe and their ceilings given $E[\bar{H}] = h_0$ to achieve $P_B = 10^{-4}$, for the case of truncated Gaussian-distributed blocking probability for each link. $\bar{N}_{max}$, defined in (3.11), is the upper bound of the expected number of paths to probe given the expected average entropy. $\bar{N}_{app}$, defined in (3.12), is approximation of $\bar{N}_{max}$. N_r is the simulated average number of paths to probe when only the average entropy is given and paths are selected randomly. N_o is the simulated average number of paths to probe when detailed blocking probability of each path is given. P_B is the target blocking probability.

probabilities $P_B = 10^{-2}$, 10^{-4}, and 10^{-6}. We observe similar results as in Figure 3-7 and 3-8. The average number of paths to probe increases linearly with $-\log_2 P_B$. N_r is bounded by $\bar{N}_{max}$ and $\bar{N}_{app}$ in a wider range of h_0 for smaller P_B due to less rounding-up effects. And N_r is always bounded by $\bar{N}_{max} + 1$ for all values of P_B.

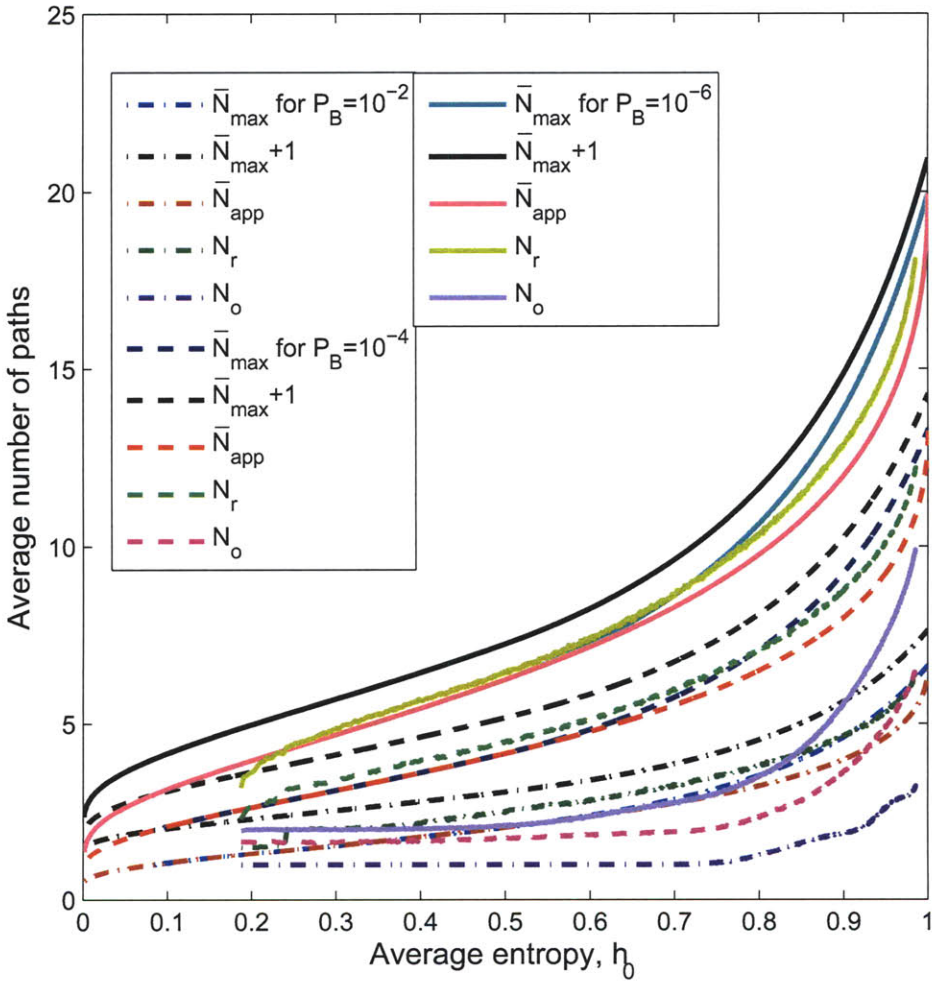

Figure 3-11: Average number of paths to probe given $E[\bar{H}] = h_0$ to achieve the target blocking probabilities $P_B = 10^{-2}$, -10^{-4}, and 10^{-6} in the simulation of truncated Gaussian-distributed blocking probability for each link. N_r is the simulated average number of paths to probe when only the average entropy is given and paths are selected randomly. N_o is the simulated average number of paths to probe when detailed blocking probability of each path is given.

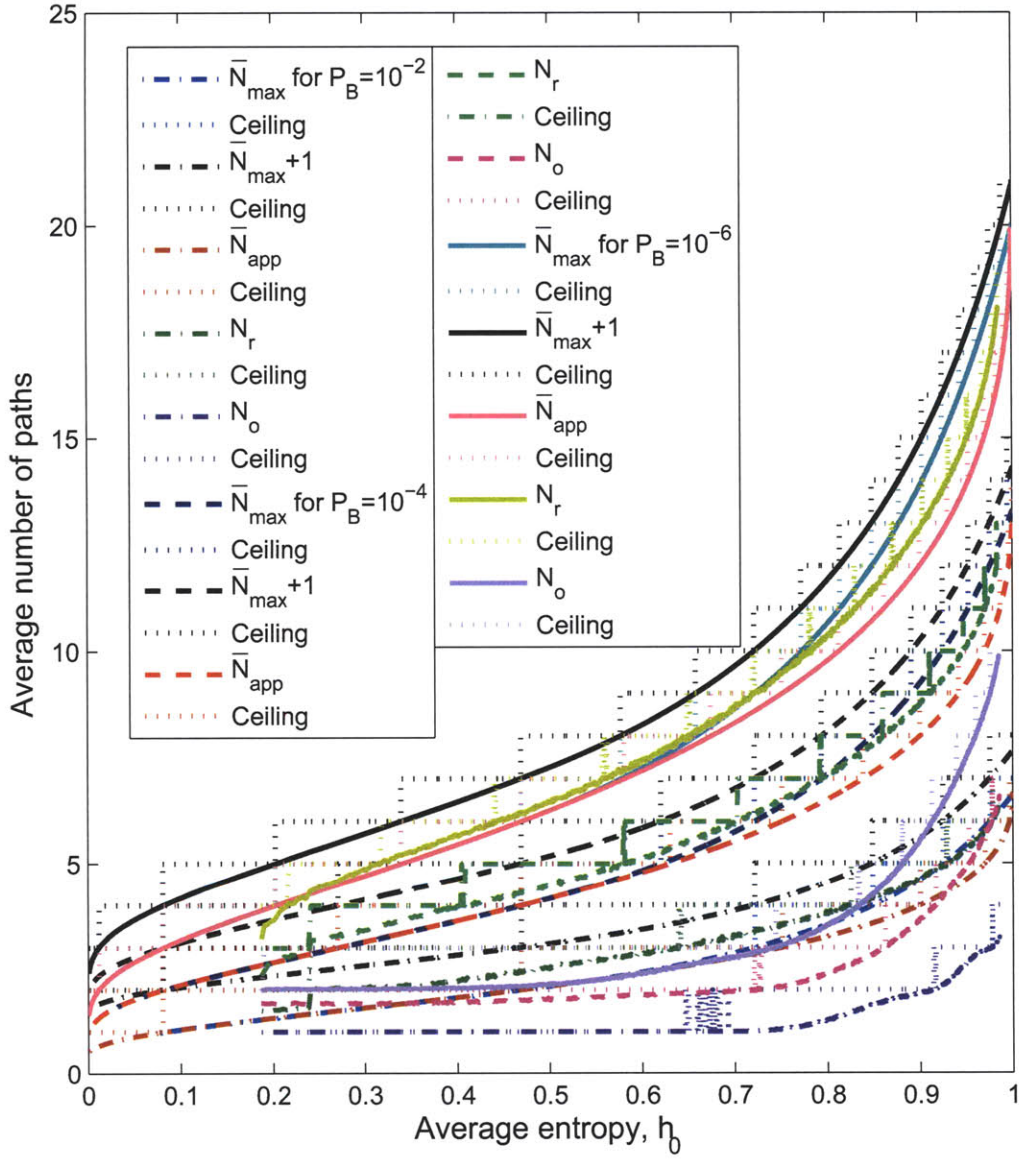

Figure 3-12: Average number of paths to probe and their ceilings given $E[\bar{H}] = h_0$ to achieve the target blocking probabilities $P_B = 10^{-2}$,$P_B = 10^{-4}$, and $P_B = 10^{-6}$ in the simulation of truncated Gaussian-distributed blocking probability for each link.

3.4 Summary of Chapter 3

In this chapter, we studied how to determine the number of paths to probe given the expected average entropy of a simple network. We formulated a Linear Programming problem and solved it to obtain the upper bound for the expected number of paths to probe. In particular, this upper bound is close to the actual average number of paths to probe in our simulations. Although we sacrifice some performance in terms of probing more paths than if we are given detailed network statistics, we have greatly reduced the amount of network management and control efforts.

Chapter 4

Fast Scheduling Algorithm for a General Network

We have studied the fast scheduling algorithm employing the probing method with the average entropy of a simple network. As shown in Figure 3-1, this simple network is composed of paths with only one link. However, real networks usually have multiple links in a path. Take an example of the optical backbone network of the United States, with its topology shown in Figure 4-1 and parameters described in Table 4.1. The US backbone network in Figure 4-1 is a mesh network composed of 60 nodes. Each link in the figure represents about 100 fiber links, and for each fiber link there are about 200 wavelength channels. The average number of hops of an end-to-end connection is four. The average node degree is 2.6, while the largest node degree is five and the least node degree is two. Therefore, a typical path in the backbone network comprises four links on the average, and there are interconnections at the connecting nodes, which introduce traffic merging and diverging. These mid-span traffic merging and diverging incur dependency among neighboring links, adding more complexity to network modeling.

For this setting, the problem arises: how can we extend our analysis in Chapter 3 to a general network which can be as complex as the one in Figure 4-1? On the one hand, we do not want to naively simplify the problem too much by assuming independencies between neighboring links, which is clearly unrealistic. On the other

hand, we do not want to introduce complicated models of the networks and traffic statistics which are intractable for a large scale network. One can suggest adding dependency among links in the modeling. This may be tractable for a network with four or five nodes, but definitely not for a large-scale network as the US backbone network. Besides, it is almost impossible for us to get a precise model of the traffic statistics in a real world large-scale network.

In this chapter, we discuss how we extend the fast-scheduling algorithm to a general network without introducing models of traffic statistics and at the same time still maintain the capability of including statistical dependencies between neighboring links. In Section 4.1, by introducing mutual information among neighboring links, we study what useful results can be drawn from Information Theory. In Section 4.2 we extend the fast-scheduling algorithm to mesh networks. We first study the method on a network with two-hop paths in Section 4.2.1. Simulation results validated the method. Then we construct a modified Bellman-Ford algorithm to pick the path that is most likely to be available from a mesh network and further extend the fast-scheduling method to a general mesh network in Section 4.2.2. Finally, we conclude this chapter in Section 4.3.

Parameter	**Value**
Number of nodes	60
Number of links	77
Average node degree	2.6
Largest node degree	5
Least node degree	2
Average link length	450 km
Number of wavelength channels per fiber link	200
Average number of hops of an end-to-end connection	4

Table 4.1: Important parameters for the US backbone network and their values, adapted from [16, Tbls. 8.1 and 8.2].

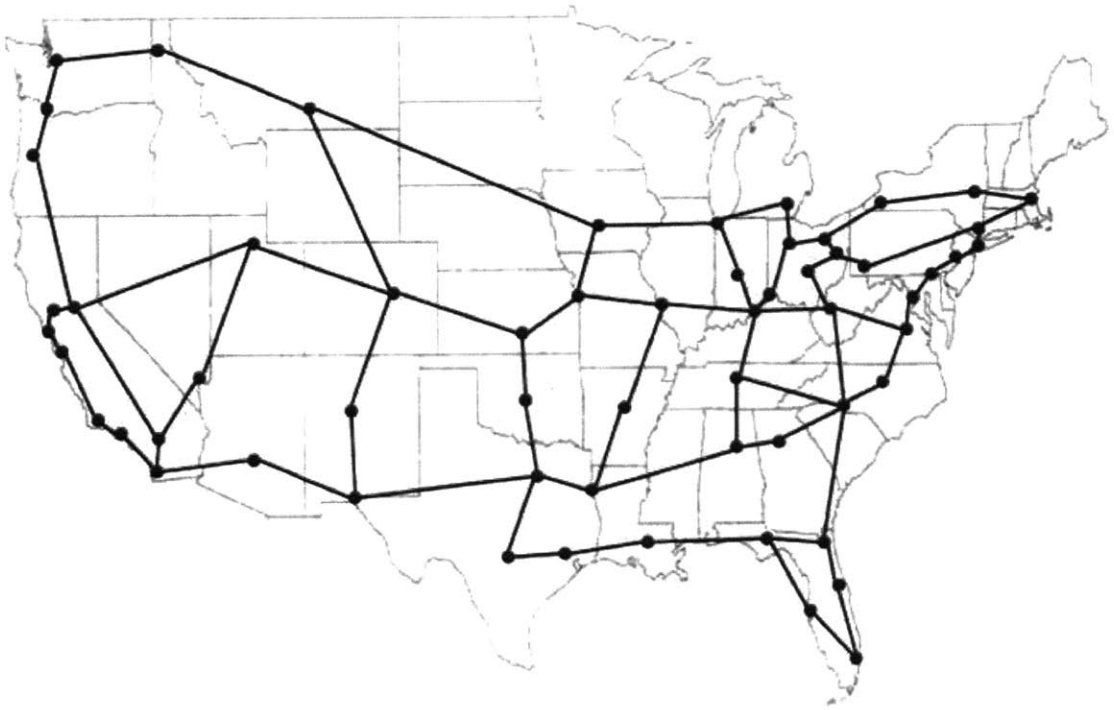

Figure 4-1: Optical backbone network of the United States. Reproduced from [16, Fig. 8.1].

4.1 Information Theoretical Analysis

Consider a path with two links L_1 and L_2 as shown in Figure 4-2. The blocking probability of link L_1 is a random variable X_1, and the blocking probability of link L_2 is a random variable X_2. For L_1, it has two states, either **0** or **1**, where **0** means that the link is available and **1** means the link is blocked. Thus, $\mathrm{P}\{L_1 = 1\} = X_1$. The entropy of L_1 can be easily calculated as $H(L_1) = H_b(X_1)$. Similarly, entropy of L_2 is $H(L_2) = H_b(X_2)$.

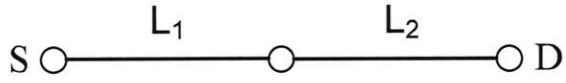

Figure 4-2: A path with two links L_1 and L_2.

From Information Theory [11], the total entropy of L_1 and L_2 is

$$H(L_1, L_2) = H(L_1) + H(L_2) - I(L_1; L_2). \tag{4.1}$$

$I(L_1; L_2)$ is the mutual information between L_1 and L_2. Given the joint probability mass function (PMF) of L_1 and L_2 ($P_{L_1L_2}$), and their marginal PMFs (P_{L_1} and P_{L_2}), $I(L_1; L_2)$ can be written as

$$I(L_1; L_2) = \sum_{L_1, L_2} (P_{L_1L_2}(l_1, l_2) \times \log_2(\frac{P_{L_1L_2}(l_1, l_2)}{P_{L_1}(l_1)P_{L_2}(l_2)})). \tag{4.2}$$

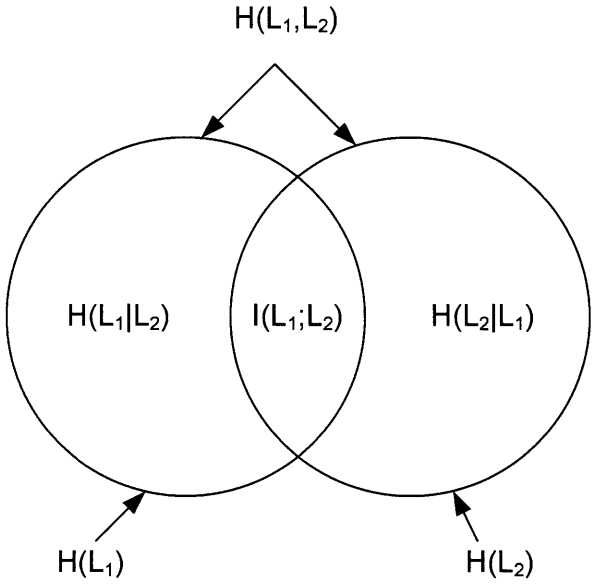

Figure 4-3: Graphical representation of entropies $H(L_1), H(L_2), H(L_1, L_2)$ and mutual information $I(L_1; L_2)$. [11, Figure 2.2.]

$I(L_1; L_2)$ can be interpreted as the correlation between L_1 and L_2. The relationship among $H(L_1), H(L_2), H(L_1, L_2)$ and $I(L_1; L_2)$ is illustrated in Figure 4-3. In the figure, the area of each circle represents the entropy of the states of each corresponding link ($H(L_1)$ or $H(L_2)$); the overlapping area represents the mutual information $I(L_1; L_2)$; and the total area circumscribed by the two circles represents the total entropy of L_1 and L_2, $H(L_1, L_2)$. There are two extreme cases for the dependency between L_1 and L_2, independent and fully dependent (correlation equals to one). For the case where L_1 and L_2 are independent, there is no overlapping between the two circles; therefore, $I(L_1; L_2)$ equals to zero. For the case where the state of L_2 can be fully determined by that of L_1, or vice versa, the two circles totally overlap with each

other; as a result, $I(L_1; L_2) = H(L_1) = H(L_2)$. For intermediate cases where the correlation between L_1 and L_2 is between zero and one, the area of overlapping increases with increase of the correlation between L_1 and L_2. Thus, $I(L_1; L_2)$ increases with increase of the correlation between L_1 and L_2.

Define a new random variable M_2 to represent the state of the whole path. M_2 is **0** if the path is available and M_2 is **1** if the path is blocked, corresponding to either L_1 or L_2 is blocked, or both of them are. Then we have the following theorem.

Theorem 4.1 *The entropy of the state of a two-hop path is smaller than or equal to the entropy of the states of its two constituent links. That is*

$$H(M_2) \leq H(L_1) + H(L_2) - I(L_1; L_2) = H(L_1, L_2). \tag{4.3}$$

Theorem 4.1 is proven in B.1. It can be easily extended to a path with three or more links in Figure 4-4. Similarly, we define a random variable M_n to represent the state of the whole path. Then we have the following theorem.

Theorem 4.2 *The entropy of the state of a n-hop path is smaller than or equal to the entropy of the states of its n constituent links. That is*

$$H(M_n) \leq \sum_{i=1}^{n} H(L_i) - \sum_{i=1}^{n-1} I(L_i; L_{i+1}). \tag{4.4}$$

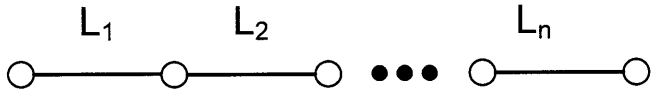

Figure 4-4: A path with n links L_1, L_2, ... , and L_n.

The proof is given in B.2.

Therefore, we can get an upper bound of the entropy of the state of a path from the entropy of states of its constituent links and the mutual information among them.

4.2 Entropy-Assisted Probing in a General Network

With the study in Section 4.1, we can obtain the upper bound of the entropy of a path. We can then use this upper bound in (3.11) to obtain an upper bound of the expected number of paths to probe. In the following analysis, we first work on a network with two-hop paths, and then extend the algorithm to a mesh network.

4.2.1 Extension to a Network with Two-Hop Paths

As discussed in Section 1.2, the evolution of entropy of groups of links and the evolution of mutual information of neighboring groups of links are broadcast periodically at the coarse time scale in the control plane. Consider the network represented in Figure 4-2, at any time t we can get a close approximation of $E[H(L_1)]$, $E[H(L_2)]$, and $E[I(L_1;L_2)]$. Assume $E[H(L_1)] = h_1$, $E[H(L_2)] = h_2$, and $E[I(L_1;L_2)] = i$, then we have $E[H(M_2)] \leq E[H(L_1) + H(L_2) - I(L_1;L_2)]$, that is, $E[H(M_2)] \leq h_1 + h_2 - i$. Substituting $h_0 = h_1 + h_2 - i$ into (3.11), we can obtain the upper bound of the average number of paths to probe for this network. In particular, larger mutual information between L_1 and L_2 corresponds to smaller upper bound of $E[H(M_2)]$, thus the upper bound of the average number of probing paths is tighter.

Simulations were carried out to test the probing method for paths with two hops. Correlation between L_1 and L_2 was introduced by defining the following conditional probability:

$$P_{X_2|X_1}(x_2|x_1) = \begin{cases} \beta & \text{if } x_2 = x_1 \\ 1-\beta & \text{if } x_2 \neq x_1 \end{cases} \tag{4.5}$$

Figure 4-5 shows the simulation results and Figure 4-6 shows the same results with ceilings. As shown in Figure 4-5, three cases with different β values were tested to achieve the same target blocking probability $P_B = 10^{-5}$. $\bar{N}_{max}$, defined in (3.11), is the upper bound of the expected number of paths to probe using $h_1 + h_2 - i$ as the upper bound of the entropy of the path. In contrast, N_1 is the simulated average

number of paths to probe for $\beta = 0.9$, N_2 is for $\beta = 0.99$ and N_3 is for $\beta = 0.999$.

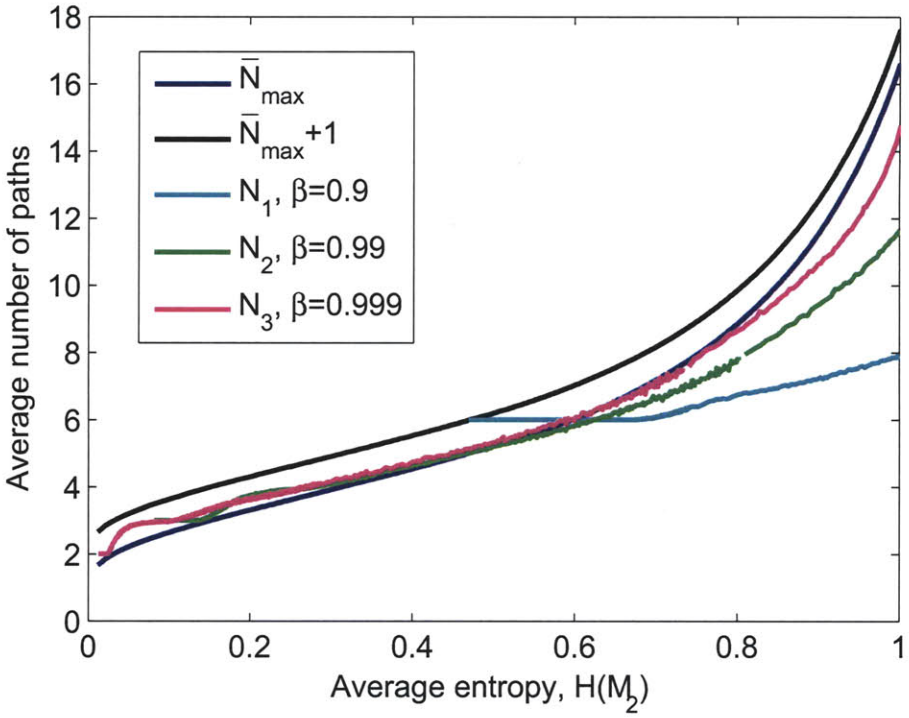

Figure 4-5: Average number of paths to probe, given H_{M2}, to achieve $P_B = 10^{-5}$ in the simulation of a network with two-hop paths. β represents the amount of dependency between the two links for one path, which is defined in (4.5). $\bar{N}_{max}$ is the theoretical upper bound for the expected number of paths to probe as defined in (3.11). N_1 is the simulation result of the actual average number of probing paths for $\beta = 0.9$. N_2 is the one for $\beta = 0.99$. N_3 is the one for $\beta = 0.999$.

As expected, $\bar{N}_{max}$ is a tight bound of N_3 and N_2 for which β equals to 0.999 and 0.99, respectively. For the cases of $\beta = 0.9$, as we lose track of the states of the network when $H(M_2)$ exceeds one, we should operate at the region where $H(M_2)$ is small. For example, even if $H(L_1) = 0$ for $\beta = 0.9$, the bound of $H(M_2)$ is $H(L_2|L_1) = H_b(0.9) = 0.469$. Therefore, the line of N_1 starts from $H(M_2) = 0.469$. For this case, we should operate at the region of $H(M_2) \in [0.47, 0.8]$, and re-broadcast the necessary information when $H(M_2)$ exceeds 0.8.

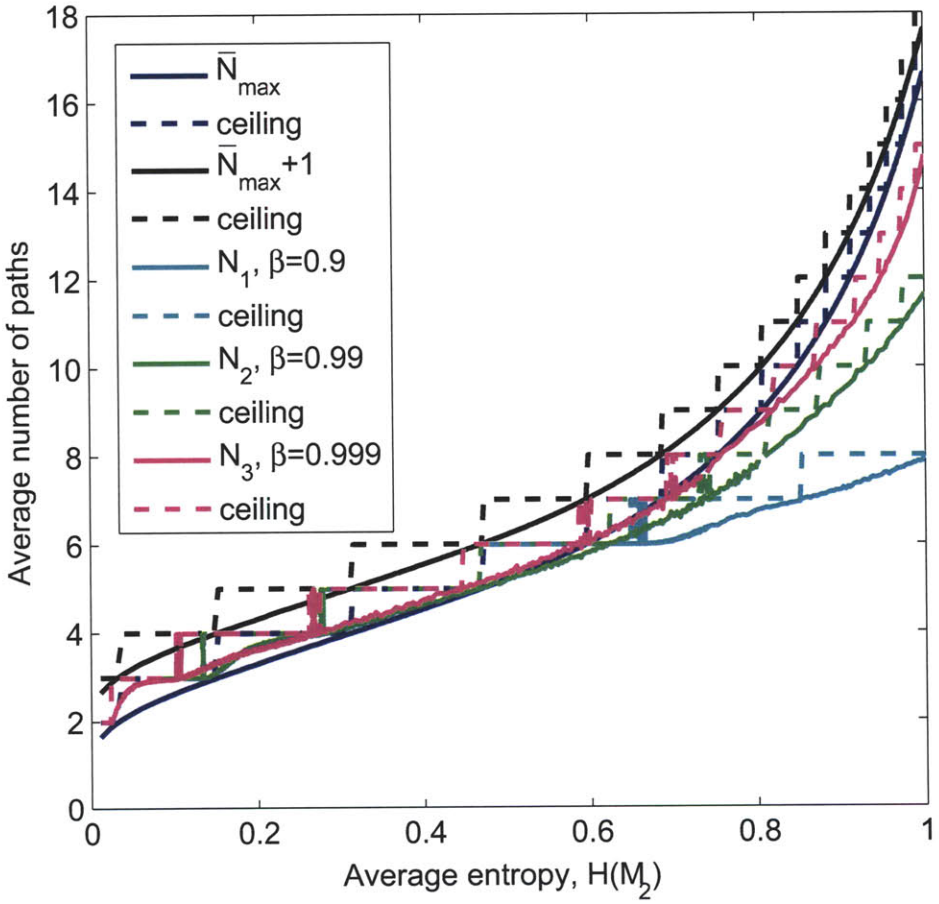

Figure 4-6: Average number of paths to probe with ceilings given H_{M2} to achieve $P_B = 10^{-5}$ in the simulation of a network with two-hop paths. β represents the amount of dependency between the two links for one path, which is defined in (4.5). $\bar{N}_{max}$ is the theoretical upper bound for the expected number of paths to probe as defined in (3.11). N_1 is the simulation result of the actual average number of probing paths for $\beta = 0.9$. N_2 is the one for $\beta = 0.99$. N_3 is the one for $\beta = 0.999$.

4.2.2 Extension to a Mesh Network

The above described method of determining the number of probing paths can be easily extended to paths with three or more links. However, in the mesh network as shown in Figure 4-7, each link in the graph can represents a group of links that connects the same pair of nodes, and there can be multiple groups of paths between the source and destination nodes. For such a case, we first need to decide which group of paths is the most likely to be available. Then we can determine the number of paths that need to be probed from that group of paths using the same method described in Section 4.2.1. The group of paths that is most likely to be open can be obtained through a modified Bellman-Ford algorithm.

Bellman-Ford algorithm [13, Chap. 5] computes shortest paths from each node to the destination node in a weighted graph with no negative cycles. The algorithm starts with searching for the shortest path from each node to the destination node with only one hop. It then iterates over the number of hops h to search for the shortest paths within h hops. It stops at the step when for each node the shortest path within $h-1$ hops is the same as the one within h hops. Define d_{ij} to be the weight of link between node **i** and node **j** and $d_{ij} = \infty$ if there is no link between them. Define D_i^h to be the length of the shortest path from node **i** to the destination node within h hops. With these definitions, the Bellman-Ford algorithm is described in Table 4.2, with node **1** being the destination node.

Algorithm	The Bellman-Ford algorithm
Initialize	
	$D_1^h = 0$ for all h
	$D_i^1 = d_{i1}$ for all $i \neq 1$
Repeat	
	$D_i^{h+1} = \min_j [d_{ij} + D_j^h]$
Until	$D_i^h = D_i^{h-1}$ for all i

Table 4.2: The Bellman-Ford algorithm.

Take the mesh network in Figure 4-7 for example, we want to find the path with the smallest expected average entropy from the source node **S** to the destination node **D**. Inspired by the Bellman-Ford algorithm, the length of each link in the graph is

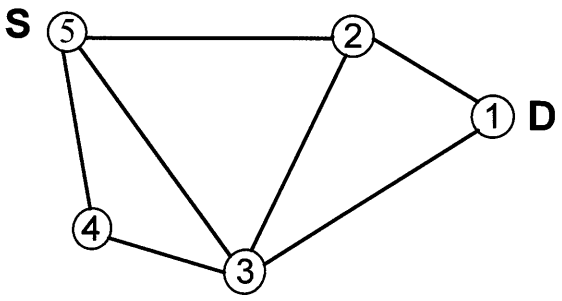

Figure 4-7: A mesh network.

taken as the expected average entropy , that is, $d_{ij} = E[H_{ij}]$. $d_{ij} = \infty$ if there is no link between node **i** and node **j**. Similarly, D_i^h is defined as the length of the shortest path from node **i** to node **1** within h hops. However, the Bellman-Ford algorithm cannot be directly applied here because the mutual information should be deducted from the neighboring links for any path and this cannot be pre-assigned into the weights of links before the shortest path is determined. Instead, the mutual information should be taken into account at each iteration step. Assuming we have already obtained D_j^h at step h, then at each node **i**, D_i^{h+1} can be obtained by selecting the shortest path among $d_{ij} + D_j^h - E[I_{ijk}]$, where k is the node to which node j is linked to in D_j^h, and I_{ijk} is the mutual information between Link ij and Link jk. The value of $d_{ij} + D_j^h - E[I_{ijk}]$ is always non-negative since mutual information is always smaller than or equal to the entropy of each link. Therefore, the path with the smallest expected average entropy from the source node **S** to the destination node **D** in Figure 4-7 can be obtained by the modified Bellman-Ford algorithm, described in Table 4.3.

After running the algorithm to find the shortest path from source to destination, we can take the "length" (as defined in the modified Bellman-Ford algorithm) of the shortest path between them as the approximation of the upper bound of the average entropy. Then we can determine how many paths we need to probe along the shortest path using (3.4).

Algorithm	The modified Bellman-Ford algorithm
Initialize	
	$D_1^h = 0$ for all h
	$D_i^1 = d_{i1}$ for all $i \neq 1$
Repeat	
	$D_i^{h+1} = \min_j[d_{ij} - E[I_{ijk}] + D_j^h]$ for all $i \neq 1$ and $h > 0$,
	where k is the node to which node j is linked to in D_j^h
	and I_{ijk} is the mutual information between Link ij and Link jk
Until	$D_i^h = D_i^{h-1}$ for all i

Table 4.3: The modified Bellman-Ford algorithm.

4.3 Summary of Chapter 4

In this chapter, we discussed how to extend the fast-scheduling algorithm to a general network. Mutual information among neighboring links was introduced to capture the dependencies among them. With that, we were able to extend the fast-scheduling algorithm to general mesh networks without introducing models of traffic statistics. Simulation results were shown to verify the method for a network with two-hop paths. A modified Bellman-Ford algorithm was designed to extend the fast-scheduling method from a simple network to a general mesh network.

Note that the algorithm discussed in Section 4.2.2 did not assume any specific network topology or models of traffic statistics. It is applicable to any network configurations, such as the US backbone network in Figure 4-1 or multiple-domain networks. This is because we have relaxed the requirement of network statistics to only the average entropy and average mutual information between neighboring links. Therefore, our algorithm is universal and robust against network models and traffic statistics with only a small sacrifice in performance.

Chapter 5

Measurement of Information of Network States

In the preceding chapters we have discussed the fast scheduling algorithm for OFS utilizing the probing method with the assistance of the entropies of network domains. We assumed that the entropies and mutual information can be estimated from sampling the states of the networks. In this chapter we discuss the methods to measure the required information: $E[H(t)]$, the expected entropy evolution of each link, and $E[I(t)]$, the expected time evolution of mutual information of two neighboring links.

5.1 Measuring the Entropy H(t)

As $H(t)$ starts from zero at time zero, we work on a set of paths that are all available at $t = 0$. We continue working on the same set to measure how the number of blocked paths increases with time, and thus we can find $H(t)$.

We assume there are many links between two nodes. The blocking probability for each link is an independently and identically distributed random variable X, as shown in Figure 3-1. To gather the necessary statistics for $H(t)$, we sample the network periodically with a time interval τ. Suppose at time zero we sample each path of the network, and from the sample results we can divide those paths into two sets, $\mathcal{A}$ and $\mathcal{B}$. $\mathcal{A}$ is the set of available paths. $\mathcal{B}$ is the set of blocked paths. We work

on $\mathcal{A}$ for the following sampling epochs. At the ith sampling epoch from $t = 0$, the number of occupied paths in $\mathcal{A}$ are noted as $N_b(i\tau)$. Then the blocking probability $X(i\tau)$ can be estimated from the following theorem.

Theorem 5.1 *Given there are $N_b(i\tau)$ number of blocked paths in $\mathcal{A}$ at the ith sampling epoch, the maximum likelihood estimate of the blocking probability at the ith sampling epoch is*

$$\hat{X}(i\tau) = \frac{N_b(i\tau)}{N(\mathcal{A})}. \tag{5.1}$$

Theorem 5.1 is proved in C.1. Therefore, entropy at the epoch $i\tau$ can be obtained as

$$H(i\tau) = H_b(\hat{X}(i\tau)). \tag{5.2}$$

The $H(t)$ obtained in this way is noisy as we only sample paths from one set $\mathcal{A}$. The fluctuations can be averaged out by taking a running time average of $H(t)$ over a period less than the coherence time of the traffic statistics. We define another sampling time interval δ, and for every δ seconds, we start with a new set $\mathcal{A}$ of available paths, and keep sampling it to get $H(t)$. Using $H_j(t)$ for the jth $H(t)$ obtained starting from $t = j\delta$, $\bar{H}(t)$ can be obtained by averaging the past k $H(t)$'s:

$$\bar{H}(t) = \frac{\sum_{j=0}^{k-1} H_j(t - j\delta)}{k}. \tag{5.3}$$

From the Strong Law of Large Number [2, Chap. 5],

$$\mathbf{P}\left(\lim_{k\to\infty} \frac{\sum_{j=0}^{k-1} H_j(t - j\delta)}{k} = E[H(t)]\right) = 1. \tag{5.4}$$

Thus, assuming the number of sampled $H(t)$ we take the average over, k, is large enough, we can approximate $E[H(t)]$ by $\bar{H}(t)$.

We also assume the length of the averaging period $k\delta$ is much smaller than the coherence time of $H(t)$ (e.g., less than one tenth of it) so that $\bar{H}(t)$ is a good predication for the $\bar{H}(t)$ in the next broadcast interval.

5.2 Measuring the Mutual Information $I(t)$

$I(t)$ can be obtained in a similar way. We work on a set of neighboring links that are all available at time zero and continue working on the same set to measure how the connectivity of these paths changes with time, and thus we can find $I(t)$. The only difference is, instead of sampling the availability of each link, we need to sample the traffic configurations at the node connecting two neighboring links.

Take for example the network fragment shown in Figure 5-1 (a). L_G, L_R and L_B with subscripts G, R, and B are used to indicate the green, red and blue link groups (no relations to the color of the wavelength) in the figure. In the following discussion, we show how the mutual information between groups of links L_G and L_R, $I_{GR}(t)$, is estimated. Define a random variable Y_G (Y_R) to denote the state of a link in the link group L_G (L_R). Y_G equals to **1** if the link is blocked, and **0** if it is available. $I_{GR}(t)$ is function of the joint PMF , P_{Y_G,Y_R}, of Y_G and Y_R. At time zero, we sample the node in Figure 5-1 (a) and note down the sets of available links in L_G/L_R as $\mathcal{A}_G/\mathcal{A}_R$. If $\mathcal{A}_G$ and $\mathcal{A}_R$ are not of the same size, then randomly drop some links in the larger set to make their sizes equal, so that $N(\mathcal{A}_G) = N(\mathcal{A}_R)$. Continue to sample these two sets periodically at time interval τ and record the traffic configurations defined in the following parameters:

$N_{11}(i\tau)$: the number of links in $\mathcal{A}_G$ that are serving traffic going through both L_G and L_R

$N_{10}(i\tau)$: the number of occupied links in $\mathcal{A}_G$ that are serving traffic going through L_G but NOT L_R

$N_{01}(i\tau)$: the number of occupied links in $\mathcal{A}_R$ that are serving traffic going through L_R but NOT L_G

$N_{00}(i\tau)$: the remaining number of links that are available in both $\mathcal{A}_G$ and $\mathcal{A}_R$, which equals to $N(\mathcal{A}_G) - N_{11}(i\tau) - N_{10}(i\tau) - N_{10}(i\tau)$

For example, for the traffic configuration shown in Figure 5-1 (b), we have $N_{11} = 2$, $N_{10} = 2$, $N_{01} = 1$ and $N_{00} = 7$.

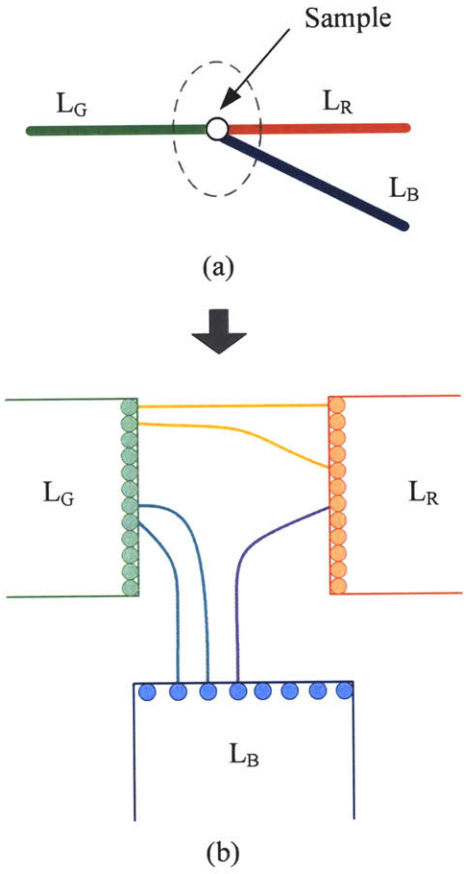

Figure 5-1: Sampling of traffic configurations to estimate $I(t)$. (a) Sampling at a node connecting three links. (b) The microscopic view of the traffic configuration at the sampling point at the node.

Using the maximum likelihood estimate, we estimate the joint PMF P_{Y_G,Y_R} by its empirical distribution, that is,

$$\hat{P}_{Y_G,Y_R}(i\tau)(y_G, y_R) = \frac{N_{Y_G Y_R}(i\tau)}{N(\mathcal{A}_G)}. \tag{5.5}$$

As such, the joint information at the epoch $i\tau$ can be obtained as

$$I_{GR}(i\tau) = \sum_{Y_G,Y_R \in 0,1} \left(\hat{P}_{Y_G,Y_R}(i\tau)(y_G, y_R) \cdot \log_2 \frac{\hat{P}_{Y_G,Y_R}(i\tau)(y_G, y_R)}{\hat{P}_{Y_G}(i\tau)(y_G)\hat{P}_{Y_R}(i\tau)(y_R)} \right), \tag{5.6}$$

where $\hat{P}_{Y_G}(i\tau)$ and $\hat{P}_{Y_R}(i\tau)$ are the marginal PMFs of Y_G and Y_R obtained from $\hat{P}_{Y_G,Y_R}(i\tau)$.

Similarly to $H(t)$, the $I(t)$ obtained from one pair of available sets for two neighboring groups of links (e.g., $\mathcal{A}_G$ and $\mathcal{A}_R$) is noisy. We should take a running time average of $I(t)$ over a period less than the coherence time of the traffic statistics to average out the fluctuations in $I(t)$. Similarly, every δ time, we start with a new pair of available sets, and keep sampling it to get $I(t)$. Using $I_j(t)$ for the jth $I(t)$ obtained starting from $t = j\delta$, $\bar{I}(t)$ can be obtained by averaging the past k $I(t)$'s:

$$\bar{I}(t) = \frac{\sum_{j=0}^{k-1} I_j(t - j\delta)}{k}. \tag{5.7}$$

Again, from the Strong Law of Large Number,

$$\mathbf{P}\left(\lim_{k\to\infty} \frac{\sum_{j=0}^{k-1} I_j(t - j\delta)}{k} = E[I(t)] \right) = 1. \tag{5.8}$$

Therefore, we can obtain a close approximation of $E[I(t)]$ by have a large enough k. We also assume the length of the averaging period $k\delta$ is much smaller than the coherence time of $I(t)$ so that $\bar{I}(t)$ is a good predication for the $\bar{I}(t)$ in the next broadcast interval.

5.3 Summary of Chapter 5

In this chapter, we studied how the required information, $E[H(t)]$ and $E[I(t)]$, are measured from online networks.

To obtain $E[H(t)]$, links in a network are sampled periodically. The link blocking probability at each sampling epoch is estimated using its empirical value. With this estimated blocking probability, binary entropy at each sampling epoch can be calculated. A running time average of $H(t)$ is taken to obtain $E[H(t)]$.

To obtain $E[I(t)]$, traffic configurations are sampled periodically at the nodes in the network periodically. The joint PMF of the two random variables representing the states of two neighboring links is estimated by its corresponding empirical values at each sampling epoch. The mutual information is obtained from the estimated joint PMFs. A running time average of $I(t)$ is taken to obtain $E[I(t)]$.

We assume the length of the averaging period for both $E[H(t)]$ and $E[I(t)]$ is much smaller than their coherence times so that $E[H(t)]$ and $E[I(t)]$ are good approximations for those in the next broadcast interval.

Chapter 6

Conclusion

With the invention and development of optical fibers in the 1970s, they have been used as a replacement for copper links to provide a tremendous amount of bandwidth, about 30 THz per fiber. However, as the network architecture still uses features that are mostly optimized for traditional electronics communications and switching, the capacity provided by optical networks has been constrained by the speed and the cost of electronics at network nodes. Therefore, it is far less than fully-utilized. In the past decade, the OFS architecture has been studied to build an all-optical data plane to provide an end-to-end, cost-effective data transport to users with large transactions. In spite of the low-cost service provided to high-end users and the relief of MAN/WAN IP routers from large transactions, there is an inherent constraint carried by the scheduling process of OFS: the possible long delays when users are scheduled to wait in a queue at the entrance of the network.

In this thesis, we present a fast scheduling algorithm for OFS employing the probing approach to set up an end-to-end connection with slightly more than one round-trip delay time.

6.1 Summary of Contributions

In the introductory chapter of this thesis, we briefly described the normal scheduling algorithm for OFS and the rather fast connection setup method employing the prob-

ing approach by others previously. We presented the motivations of our study of a new fast scheduling algorithm for OFS, and outlined how our algorithm works. In particular, we explained why our algorithm can greatly reduce the amount of sensing and information dissemination by the network management and control system, and why it is robust to network models and traffic statistics.

Chapter 2 demonstrated entropy evolution of a single link, and entropy evolution of a single path with L independent links. For both cases, with the initial state of the link (or path) known, as time passes, its entropy either increases to the maximum and stays there, or first increases to the maximum and then decreases to its steady state value. With the same traffic statistics at each link, entropy of a path with more than one independent link increases faster than entropy of a single link. If the correlations between any two neighboring links of a path are smaller than one, this can be considered as the intermediate case between two extremities of a single link and a path with L independent links, of which the entropy also increases from zero as time passes. The average entropy evolution also starts from zero and keeps increasing within one broadcast interval.

Chapter 3 provided a detailed analysis on how to determine the number of paths to probe if we are given the expected average entropy of a simple network. A Linear Programming problem was formulated and solved to obtain an upper bound for the expected number of paths. In particular, this upper bound turned out to be very close to the actual average number of paths to probe in our simulations. Although some performance is sacrificed in terms of probing more paths than if detailed network statistics are given, the amount of network management and control information passed and processed is greatly reduced.

Chapter 4 presented how to extend the fast-scheduling algorithm to a general network. Mutual information among neighboring links was introduced to capture any dependencies among them. With that, the fast-scheduling algorithm was extended to general mesh networks without introducing models of traffic statistics. Simulation results were shown to verify the method for a network with two-hop paths. A modified Bellman-Ford algorithm was designed to extend the fast-scheduling method from a

simple network with only one source-destination pair to a general mesh network.

Lastly, Chapter 5 showed how the required information, $E[H(t)]$ and $E[I(t)]$, are collected from online networks.

6.2 Future Work and Challenges

One challenge will be the fine-tuning of our algorithm for heterogeneous networks. Since for different kinds of networks, they have different intrinsic properties. For example, there are fading problems in free-space optical links, multi-path problems for wireless radio frequency links, while fiber links are much more reliable and robust. The entropy for wireless networks may increase much faster than that of the optical fiber networks. The mutual information between two such heterogeneous networks might not be the best candidate to take into account this vast difference in properties between them. Instead, some additional parameter might be introduced to better model the heterogeneity.

Another direction for future work lies in the incorporation of our algorithm, which is blind to network models and traffic statistics, with some special information of traffic statistics. For example, if somehow we can predict the traffic arrival and departure processes in the network are Poisson processes, we should exploit this information to improve the performance of our algorithm. For regions with predictable traffic patterns, a hybrid algorithm should be able to give a better performance.

Appendix A

Proofs and Derivations for Chapter 3

A.1 Proof of Theorem 3.1

Theorem 3.1 is restated below:

Theorem *Let X_c be a continuous random variable with probability density function $f_{X_c}(x) \in \mathcal{C}$. Let X_d^* be the discrete random variable that achieves optimum solution in Problem 3.4. Then, $E[-\log_2(X_c)] \geq E[-\log_2(X_d^*)]$.*

The proof is done by contradiction.

Proof: Assume $E[-\log_2(X_c)] < E[-\log_2(X_d^*)]$ and $\epsilon = E[-\log_2(X_d^*)] - E[-\log_2(X_c)]$. Let Y_n be a sequence of discrete random variables defined by

$$Y_n = \frac{\lfloor nX_c \rfloor}{n}, \qquad \text{for } n = \{1, 2, ...\}.$$

Then, we have $Y_n \leq Y_{n+1}$ for any $n \geq 1$ and

$$Y_n \to X_c \text{ almost surely.} \tag{A.1}$$

Since $\log_2(x)$ and $H_b(b)$ are both monotonic functions of x for $x \in (0, 1/2)$, by mono-

tone convergence theorem [26, Chapter 5], we have

$$E[-\log_2(Y_n)] \rightarrow E[-\log_2(X_c)], \text{ and} \tag{A.2}$$

$$E[H_b(Y_n)] \rightarrow E[H_b(X_c)] = h_0. \tag{A.3}$$

Now, define another sequence of discrete random variables

$$\hat{Y}_n^{\alpha} = Y_n + (\frac{1}{2} - Y_n)\alpha. \tag{A.4}$$

Let $g_n(\alpha) = E[H_b(\hat{Y}_n^{\alpha})]$. Then $g_n(\alpha)$ is a continuous and monotonically increasing function of α, and we have

$$g_n(0) = E[H_b(Y_n)], \text{ and} \tag{A.5}$$

$$g_n(1) = E[H_b(\frac{1}{2})] = 1. \tag{A.6}$$

Because $Y_n \leq X_c$, $g_n(0) = E[H_b(Y_n)] \leq h_0$. Therefore, there exists $\alpha_n^* \in [0,1]$, such that $g_n(\alpha_n^*) = h_0$. However, as $g_n(0) = E[H_b(Y_n)] \rightarrow E[H_b(X_c)] = h_0$, we have $\alpha_n^* \rightarrow 0$.

Therefore, we have

$$E[-log_2(\hat{Y}_n^{\alpha_n^*})] \rightarrow E[-\log_2(Y_n)], \tag{A.7}$$

which implies

$$E[-log_2(\hat{Y}_n^{\alpha_n^*})] \rightarrow E[-\log_2(X_c)]. \tag{A.8}$$

Now we have constructed a sequence of discrete random variables $\hat{Y}_n^{\alpha_n^*}$ that converges to X_c. Therefore, we can find an N such that

$$E[-log_2(\hat{Y}_N^{c_N^*})] < E[-\log_2(X)] + \epsilon = E[-\log_2(X_d^*)]. \tag{A.9}$$

But $\hat{Y}_N^{c_N^*}$ is a discrete random variable, so we must have $E[-log_2(\hat{Y}_N^{c_N^*})] \geq E[-\log_2(X_d^*)]$, leading to contradiction of the assumption. Q.E.D.

A.2 Derivation of (3.10)

Equation (3.10) is the solution to Problem 3.5, which is equivalent to Problem 3.3, in Section 3.2. We restate Problem 3.5 below.

Problem 3.5 *The minimum of* $E[-\log_2(X)]$ *over* $f_X(x) \in \mathcal{C}$ *is determined by*

$$\begin{aligned} E[-\log_2(X)]_{min} = \min_{\alpha\in[0,1],x_1,x_2\in[0,0.5]} & -\alpha\log_2(x_1) - (1-\alpha)\log_2(x_2) \\ \textit{subject to:} \quad & -\alpha H_b(x_1) - (1-\alpha)H_b(x_2) = h_0. \end{aligned} \tag{A.10}$$

Next we are going to show that the solution to Problem 3.5 is given in (3.10), which is re-illustrated here:

$$E[-\log_2(X)]_{min} = \begin{cases} -\log_2[H_b^{-1}(h_0)] & \text{if } h_0 \le h_A \\ \frac{(1-h_0)\{-\log_2[H_b^{-1}(h_0)]\}+h_0-h_A}{1-h_A} & \text{if } h_0 > h_A \end{cases},$$

where $H_b^{-1}(h_0)$ is the inverse function of $H_b(x) = h_0$ for $x \in (0, 0.5)$. h_A is the solution to $\frac{h_A-1}{H_b^{-1}(h)\log 2\cdot\log_2\frac{1-H_b^{-1}(h)}{H_b^{-1}(h)}} - log_2 H_b^{-1}(h) - 1 = 0$ and, numerically, $h_A \approx 0.4967$.

Proof: First we transform (A.10) into:

$$E[-\log_2(X)]_{min} = \min_{\alpha\in[0,1],h_1,h_2\in[0,1)} -\alpha\log_2(H_b^{-1}(h_1)) - (1-\alpha)\log_2(H_b^{-1}(h_2)) \tag{A.11}$$

$$\text{subject to: } \alpha h_1 + (1-\alpha)h_2 = h_0. \tag{A.12}$$

Define function $f(h)$ to be

$$f(h) = -\log_2(H_b^{-1}(h)) \text{ for } h \in (0,1). \tag{A.13}$$

Then,

$$\begin{aligned}\frac{df}{dh} &= -\frac{1}{(\log 2)\cdot H_b^{-1}(h)\cdot \log_2 \frac{1-H_b^{-1}(h)}{H_b^{-1}(h)}} \\ &= -\frac{1}{(\log 2)x\log_2 \frac{1-x}{x}}, \text{ and}\end{aligned} \tag{A.14}$$

$$\begin{aligned}\frac{d^2f}{dh^2} =& \frac{(1-H_b^{-1}(h))\log_2 \frac{1-H_b^{-1}(h)}{H_b^{-1}(h)} - \frac{1}{\log 2}}{(\log 2)(H_b^{-1}(h))^2(1-H_b^{-1}(h))(\log_2 \frac{1-H_b^{-1}(h)}{H_b^{-1}(h)})^3} \\ =& \frac{(1-x)\log_2 \frac{1-x}{x} - \frac{1}{\log 2}}{(\log 2)x^2(1-x)(\log_2 \frac{1-x}{x})^3},\end{aligned} \tag{A.15}$$

where $x = H_b^{-1}(h)$, and $x \in (0, 0.5)$.

From the above two equations, we can show $\frac{df}{dh}$ is smaller than zero for $h \in (0,1)$, and $\frac{d^2f}{dh^2}$ is a monotonically decreasing function for $h \in (0,1)$. Figure A-1 plots $\frac{df}{dh}$ and $\frac{d^2f}{dh^2}$ with respect to h. From the figure we can see $\frac{df}{dh}$ first increases to its maximum at h_C, and then decreases for $h > h_C$, where h_C can be solved from $(1-H_b^{-1}(h))\log_2 \frac{1-H_b^{-1}(h)}{H_b^{-1}(h)} - \frac{1}{\log 2} = 0$ and numerically, $h_C \approx 0.7561$.

Therefore, $f(h)$ is convex for $h \in (0, h_C)$, and concave for $h \in (h_C, 1)$. $f(h)$ is plotted in Figure A-2 and Line AB is the tangent line of $f(h)$ that passes points B (1,1) and A (h_A, $f(h_A)$). Realizing (A.11) and (A.12) are linear combinations of either $f(h)$ or h of the same ratio, it is clear that when $h_0 < h_A$, the minimum happens exactly at point $(h_0, f(h_0))$, and when $h_0 > h_A$, the minimum falls on the tangent line AB (See Figure A-2).

To solve for h_A, we define the slope of the line passing through $(h, f(h))$ and B(1,1) to be k, that is

$$k = \frac{-\log_2[H_b^{-1}(h)] - 1}{h-1}. \tag{A.16}$$

Taking the first derivative of k with respect to h, we have

$$\frac{dk}{dh} = \frac{-\frac{h-1}{H_b^{-1}(h)\log(2)\cdot\log_2(\frac{1-H_b^{-1}(h)}{H_b^{-1}(h)})} + \log_2(H_b^{-1}(h)) + 1}{(h-1)^2}. \tag{A.17}$$

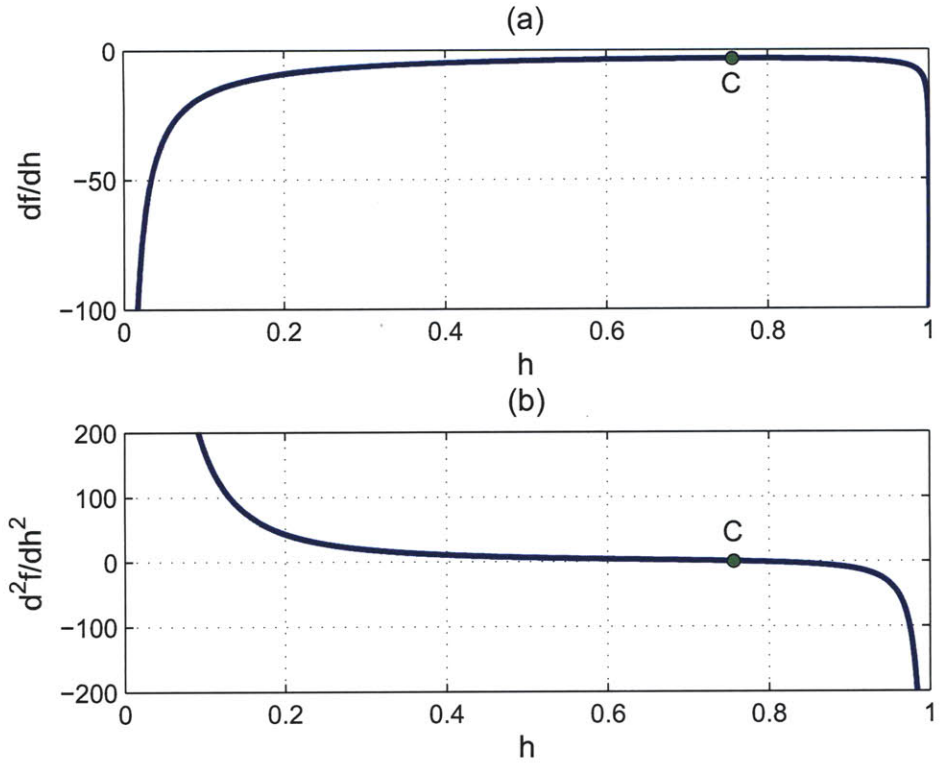

Figure A-1: The first and second derivatives of $f(h)$ with respect to $h \in (0,1)$. d^2f/dh^2 equals to zero at point C. (a) df/dh. (b) d^2f/dh^2.

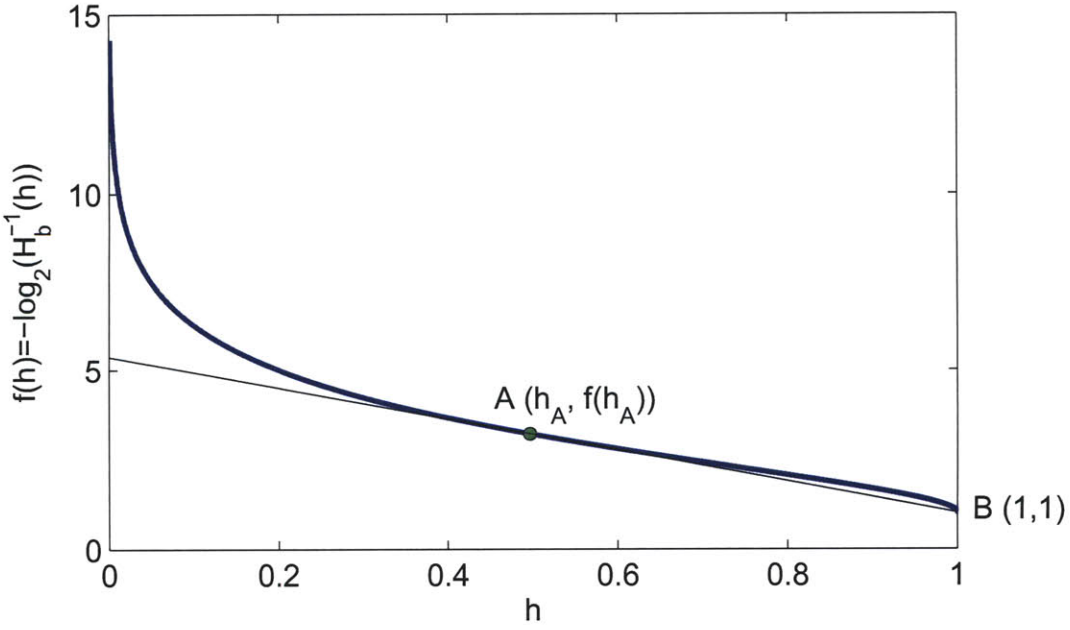

Figure A-2: $f(h)$ with respect to $h \in (0,1)$. Line AB is the tangent line of $f(h)$ that passes B.

h_A can be solved by letting $\frac{dk}{dh}$ equal to zero, that is

$$\frac{h-1}{H_b^{-1}(h)\log 2 \cdot \log_2 \frac{1-H_b^{-1}(h)}{H_b^{-1}(h)}} - log_2 H_b^{-1}(h) - 1 = 0. \tag{A.18}$$

Numerically, $h_A \approx 0.4967$.

Since Line AB can be expressed as $\frac{(1-h_0)\{-\log_2[H_b^{-1}(h_0)]\}+h_0-h_A}{1-h_A}$, the solution to Problem 3.5 is

$$E[-\log_2(X)]_{min} = \begin{cases} -\log_2[H_b^{-1}(h_0)] & \text{if } h_0 \leq h_A \\ \frac{(1-h_0)\{-\log_2[H_b^{-1}(h_0)]\}+h_0-h_A}{1-h_A} & \text{if } h_0 > h_A \end{cases},$$

Q.E.D.

Appendix B

Proofs and Derivations for Chapter 4

B.1 Proof of Theorem 4.1

Theorem 4.1 is restated here:

Theorem *The entropy of the state of a two-hop path is smaller than or equal to the entropy of the states of its two constituent links. That is*

$$H(M_2) \leq H(L_1) + H(L_2) - I(L_1; L_2) = H(L_1, L_2). \tag{B.1}$$

Proof: The right hand side of (B.1) can be written as

$$H(L_1, L_2) = -\sum_{x_1 \in \{0,1\}} \sum_{x_2 \in \{0,1\}} P_{x_1,x_2}(x_1, x_2) \log_2(P_{x_1,x_2}(x_1, x_2)), \tag{B.2}$$

The left hand side $H(M_2)$ can be written as:

$$\begin{aligned} H(M_2) = & - P_{x_1,x_2}(0,0) \cdot \log_2(P_{x_1,x_2}(0,0)) \\ & - (P_{x_1,x_2}(0,1) + P_{x_1,x_2}(1,0) + P_{x_1,x_2}(1,1)) \cdot \\ & \log_2(P_{x_1,x_2}(0,1) + P_{x_1,x_2}(1,0) + P_{x_1,x_2}(1,1)) \end{aligned} \tag{B.3}$$

Define function $g(x) = -x \log_2 x$. Since

$$g''(x) = -\frac{1}{x \log 2}, \tag{B.4}$$

$g''(x) < 0$ for $x \in (0, 0.5)$. Thus $g(x)$ is convex for $x \in (0, 0.5)$. As a result, we have

$$\begin{aligned} -P_{x_1,x_2}(1,0) \cdot \log_2(P_{x_1,x_2}(1,0)) \leq & - (P_{x_1,x_2}(1,0) + P_{x_1,x_2}(0,1)) \cdot \\ & \log_2(P_{x_1,x_2}(1,0) + P_{x_1,x_2}(0,1)) \\ & - [-P_{x_1,x_2}(0,1) \cdot \log_2(P_{x_1,x_2}(0,1))], \end{aligned} \tag{B.5}$$

with equality when $P_{x_1,x_2}(1,0) = 0$.

$$\begin{aligned} -P_{x_1,x_2}(1,1) \cdot \log_2(P_{x_1,x_2}(1,1)) \leq & - (P_{x_1,x_2}(1,0) + P_{x_1,x_2}(0,1) + P_{x_1,x_2}(1,1)) \cdot \\ & \log_2(P_{x_1,x_2}(1,0) + P_{x_1,x_2}(0,1) + P_{x_1,x_2}(1,1)) \\ & - [-(P_{x_1,x_2}(1,0) + P_{x_1,x_2}(0,1)) \cdot \\ & \log_2(P_{x_1,x_2}(1,0) + P_{x_1,x_2}(0,1))], \end{aligned} \tag{B.6}$$

with equality when $P_{x_1,x_2}(1,1) = 0$.

Summing up (B.5) and (B.6), we obtain

$$\begin{aligned} -[P_{x_1,x_2}(0,1) + P_{x_1,x_2}(1,0) + P_{x_1,x_2}(1,1)] \cdot \log_2[P_{x_1,x_2}(0,1) + P_{x_1,x_2}(1,0) + P_{x_1,x_2}(1,1)] \\ \leq - \sum_{(x_1,x_2) \neq (0,0)} \{P_{x_1,x_2}(x_1,x_2) \log_2[P_{x_1,x_2}(x_1,x_2)]\}, \end{aligned}$$

with equality when only one of the

$$P_{x_1,x_2}(x_1,x_2) \in \{P_{x_1,x_2}(x_1,x_2); x_1 \in \{0,1\}, x_2 \in \{0,1\} \text{ and } (x_1,x_2) \neq (0,0)\}$$

is not equal to zero.

Therefore, $H(M_2) \leq H(L_1) + H(L_2) - I(L_1;L_2)$.

Q.E.D.

B.2 Proof of Theorem 4.2

We restate Theorem 4.2 here:

Theorem *The entropy of the state of a n-hop path is smaller than or equal to the entropy of the states of its n constituent links. That is*

$$H(M_n) \leq \sum_{i=1}^{n} H(L_i) - \sum_{i=1}^{n-1} I(L_i;L_{i+1}). \tag{B.7}$$

Proof: The right hand side of (B.7) equals to $H(L_1, L_2, \cdots, L_n)$, which can be written as

$$\begin{aligned} H(L_1, L_2, \cdots, L_n) = - \sum_{x_1,x_2,\cdots,x_n \in \{0,1\}} & [P_{x_1,x_2,\cdots,x_n}(x_1, x_2, \cdots, x_n) \cdot \\ & \log_2 P_{x_1,x_2,\cdots,x_n}(x_1, x_2, \cdots, x_n)]. \end{aligned} \tag{B.8}$$

The left hand side of (B.7) is

$$\begin{aligned} H(M_n) = & - P_{x_1,x_2,\cdots,x_n}(0,0,\cdots,0) \cdot \log_2 P_{x_1,x_2,\cdots,x_n}(0,0,\cdots,0) \\ & - [1 - P_{x_1,x_2,\cdots,x_n}(0,0,\cdots,0)] \cdot \log_2[1 - P_{x_1,x_2,\cdots,x_n}(0,0,\cdots,0)]. \end{aligned} \tag{B.9}$$

Following the same argument as in B.1, we have $H(M_n) \leq H(L_1, L_2, \cdots, L_n)$.

Q.E.D

Appendix C

The Proof for Chapter 5

C.1 Proof of Theorem 5.1

Theorem 5.1 is restated below:

Theorem *Given there are $N_b(i\tau)$ number of blocked paths in $\mathcal{A}$ at the ith sampling epoch, the maximum likelihood estimate of the blocking probability at the ith sampling epoch is*

$$\hat{X}(i\tau) = \frac{N_b(i\tau)}{N(\mathcal{A})}. \tag{C.1}$$

Proof: The blocking probability of each path in $\mathcal{A}$ at the ith sampling epoch is an unknown parameter $X(i\tau)$. Define the random variable $N_{i\tau}$ to be the number of blocked paths in $\mathcal{A}$ at time $i\tau$. Then $N_{i\tau}$ is a binomial random variable with parameter $N(\mathcal{A})$ and $X(i\tau)$. Therefore, the likelihood that there are $N_b(i\tau)$ number of occupied paths in $\mathcal{A}$ is

$$P_{N_{i\tau}}(N_b(i\tau); X(i\tau)) = \begin{pmatrix} N(\mathcal{A}) \\ N_b(i\tau) \end{pmatrix} X(i\tau)^{N_b(i\tau)}(1 - X(i\tau))^{N(\mathcal{A})-N_b(i\tau)}. \tag{C.2}$$

Then the maximum likelihood estimate [2, Chap. 9] of $X(i\tau)$ is

$$\begin{aligned}\hat{X}(i\tau) &= \arg\max_{X(i\tau)} P_{N_{i\tau}}(N_b(i\tau); X(i\tau)) \\ &= \arg\max_{X(i\tau)} \left(\begin{array}{c} N(\mathcal{A}) \\ N_b(i\tau) \end{array} \right) X(i\tau)^{N_b(i\tau)}(1 - X(i\tau))^{N(\mathcal{A})-N_b(i\tau)}\end{aligned} \tag{C.3}$$

Taking the logarithm of the likelihood function in (C.2), we get the log-likelihood function [2, Chap. 9],

$$\begin{aligned}L &= \log P_{N_{i\tau}}(N_b(i\tau); X(i\tau)) \\ &= \log \left(\begin{array}{c} N(\mathcal{A}) \\ N_b(i\tau) \end{array} \right) + N_b(i\tau) \log X(i\tau) + (N(\mathcal{A}) - N_b(i\tau)) \log(1 - X(i\tau)).\end{aligned} \tag{C.4}$$

and from the monotonic-increasing property of the log function, we have

$$\hat{X}(i\tau) = \arg\max_{X(i\tau)} \log P_{N_{i\tau}}(N_b(i\tau); X(i\tau)). \tag{C.5}$$

Since

$$\frac{dL}{dX(i\tau)} = \frac{N_b(i\tau)}{X(i\tau)} - \frac{N(\mathcal{A}) - N_b(i\tau)}{1 - X(i\tau)}, \text{ and} \tag{C.6}$$

$$\frac{d^2L}{dX(i\tau)^2} = -\frac{N_b(i\tau)}{X(i\tau)^2} - \frac{N(\mathcal{A}) - N_b(i\tau)}{(1 - X(i\tau))^2} < 0, \tag{C.7}$$

$\hat{X}(i\tau)$ can be obtained by letting $\frac{\partial L}{\partial X(i\tau)} = 0$. Therefore, $\hat{X}(i\tau) = \frac{N_b(i\tau)}{N(\mathcal{A})}$.
Q.E.D.

Bibliography

[1] S. Alexander, R. Bondurant, D. Byrne, V. Chan, S. Finn, R. Gallager, B. Glance, H. Haus, P. Humblet, R. Jain, I. Kaminow, M. Karol, R. Kennedy, A. Kirby, H. Le, A. Saleh, B. Schofield, J. Shapiro, N. Shankaranarayanan, R. Thomas, R. Williamson, and R. Wilson. A precompetitive consortium on wide-band all-optical networks. *Lightwave Technology, Journal of*, 11(5):714–735, May-Jun 1993.

[2] D. P. Bertsekas and J. N. Tsitsiklis. *Introduction to probability.* Athena Scientific, Belmont, Mass., 2008.

[3] D. Bertsimas and J. N. Tsitsiklis. *Introduction to linear optimization.* Athena Scientific, Belmont, Mass., 1997.

[4] V. Chan. Guest editorial optical communications and networking series. *Selected Areas in Communications, IEEE Journal on*, 23(8):1441–1443, Aug. 2005.

[5] V. Chan. Optical flow switching. In *OptoElectronics and Communications Conference, 2009. OECC 2009. 14th*, pages 1–3, July 2009.

[6] V. Chan. Optical flow switching: A new "green" transport mechanism for fiber networks. In *Transparent Optical Networks, 2009. ICTON '09. 11th International Conference on*, pages 1–1, July 2009.

[7] V. Chan, A. Ganguly, and G. Weichenberg. Optical flow switching with time deadlines for high-performance applications. *IEEE Globecom 2009*, 2009.

[8] V. Chan, K. Hall, E. Modiano, and K. Rauschenbach. Architectures and technologies for high-speed optical data networks. *Lightwave Technology, Journal of*, 16(12):2146–2168, Dec 1998.

[9] V. Chan, A. Kirby, and A. Saleh. The AT&T Digital, and MIT All Optical Network Consortium. In *LEOS 1993*, pages G52 –G53, Jul 1993.

[10] V. Chan, G. Weichenberg, and M. Medard. Optical flow switching. *Broadband Communications, Networks and Systems, 2006. BROADNETS 2006. 3rd International Conference on*, pages 1 –8, Oct. 2006.

[11] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley-Interscience, Hoboken, N.J., 2006.

[12] N. M. Froberg, S. R. Henion, H. G. Rao, B. K. Hazzard, S. Parikh, B. R. Romkey, and M. Kuznetsov. The NGI ONRAMP Test Bed: Reconfigurable WDM Technology for Next Generation Regional Access Networks. *J. Lightwave Technol.*, 18(12):1697, 2000.

[13] R. G. Gallager. *Discrete stochastic processes.* Kluwer Academic Publishers, Boston, 1996.

[14] B. Ganguly. *Implementation and modeling of a scheduled Optical Flow Switching (OFS) network.* Ph.D. thesis, Massachusetts Institute of Technology, 2008.

[15] B. Ganguly and V. Chan. A scheduled approach to optical flow switching in the onramp optical access network testbed. *Optical Fiber Communication Conference and Exhibit, 2002. OFC 2002*, pages 215 – 216, Mar 2002.

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

[17] G. Weichenberg. *Design and Analysis of Optical Flow Switched Networks.* Ph.D. thesis defense, Massachusetts Institute of Technology, November 6, 2008.

[18] G. Weichenberg, V. Chan, and M. Medard. On the throughput-cost tradeoff of multi-tiered optical network architectures. In *Global Telecommunications Conference, 2006. GLOBECOM '06. IEEE*, pages 1–6, Dec 2006.

[19] G. Weichenberg, V. Chan, and M. Medard. Access network design for optical flow switching. In *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, pages 2390–2395, Nov. 2007.

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

[21] G. Weichenberg, V. Chan, and M. Medard. Performance analysis of optical flow switching. In *Communications, 2009. ICC '09. IEEE International Conference on*, pages 1–6, June 2009.

[22] G. Weichenberg, V. Chan, E. Swanson, and M. Medard. Throughput-cost analysis of optical flow switching. In *Optical Fiber Communication - includes post deadline papers, 2009. OFC 2009. Conference on*, pages 1–3, March 2009.

[23] G. Weichenberg, V. W. S. Chan, and M. Medard. Cost-efficient optical network architectures. In *Optical Communications, 2006. ECOC 2006. European Conference on*, pages 1–2, Sep. 2006.

[24] G. Weichenberg, V. W. S. Chan, and M. Medard. On the capacity of optical networks: A framework for comparing different transport architectures. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–13, April 2006.

[25] G. E. Weichenberg. *Design and analysis of optical flow switched networks.* Ph.D. thesis, Massachusetts Institute of Technology, 2009.

[26] D. Williams. *Probability with martingales.* Cambridge University Press, Cambridge, Eng. ; New York, 1991.

[27] L. Zhang and V. Chan. Fast scheduling for optical flow switching. In *IEEE Globecom 2010 - Optical Networks and Systems Symposium*, Miami, Florida, USA, 2010.