MODELING AND ANALYSIS OF THE PERFORMANCE OF NETWORKS IN FINITE-BUFFER REGIME

A Dissertation
Presented to
The Academic Faculty

By

Nima Torabkhani

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in
Electrical and Computer Engineering

School of Electrical and Computer Engineering
Georgia Institute of Technology
May 2014

MODELING AND ANALYSIS OF THE PERFORMANCE OF NETWORKS IN FINITE-BUFFER REGIME

Date Approved: January 2014

To my mother, father and my brother,

without whom this would not be a reality.

ACKNOWLEDGMENTS

Foremost, I would like to express my appreciation to my advisor Dr. Faramarz Fekri for the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, and knowledge. His guidance helped me in all the time of research and writing of this thesis. Besides my advisor, I wish to thank the rest of my committee members who were more than generous with their expertise and precious time. Thank you Dr. John Barry, Dr. Raghupathy Sivakumar, Dr. Howard Weiss, and Dr. Alenka Zajic for agreeing to serve on my committee. I would also like to thank Dr. William Trotter in School of Mathematics and Dr. Justin Romberg in School of Electrical and Computer Engineering for their support, help, suggestions and encouragement.

I thank my fellow labmates in Georgia Tech Information Processing, Communications and Security Research Lab: Ahmad Beirami, Mohsen Sardari, Arash Einolghozati, Dr. Badri Vellambi, for being supportive throughout my graduate studies and for all the fun we have had in the last six years. I also thank my friends (too many to list here but you know who you are) for providing support and friendship that I needed the most. Last but not the least, I would like to thank my family for supporting me spiritually and financially throughout my graduate school years.

This research was supported in part by the National Science Foundation, Grant No. CCF-0914630.

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY

In networks, using large buffers tend to increase end-to-end packet delay and its deviations, conflicting with real-time applications such as online gaming, audio-video services, IPTV, and VoIP. Further, large buffers complicate the design of high speed routers, leading to more power consumption and board space. According to Moore's law, switching speeds double every 18 months while memory access speeds double only every 10 years. Hence, as memory requirements increasingly become a limiting aspect of router design, studying networks in finite-buffer regime seems necessary for network engineers.

This work focuses on both practical and theoretical aspects of finite-buffer networks. In Chapters 1 – 7, we investigate the effects of finite buffer sizes on the throughput and packet delay in different networks. These performance measures are shown to be linked to the stationary distribution of an underlying irreducible Markov chain that exactly models the changes in the network. An iterative scheme is proposed to approximate the steady-state distribution of buffer occupancies by decoupling the exact chain to smaller chains. These approximate solutions are used to analytically characterize network throughput and packet delay, and are also applied to some network performance optimization problems. Further, using simulations, it is confirmed that the proposed framework yields accurate estimates of the throughput and delay performance measures and captures the vital trends and tradeoffs in these networks. In Chapters 8 – 10, we address the problem of modeling and analysis of the performance of finite-memory random linear network coding in erasure networks. When using random linear network coding, the content of buffers creates dependencies which cannot be captured directly using the classical queueing theoretical models. A careful derivation of the buffer occupancy states and their transition rules are presented as well as decodability conditions when random linear network coding is performed on a stream of arriving packets.

CHAPTER 1
INTRODUCTION AND RELATED WORK

In networks, packets have to be routed between nodes through a series of intermediate relay nodes (*i.e.*, routers). Each intermediate node in the network may receive packets via multiple data streams that are routed simultaneously from their source nodes to their respective destinations. In such conditions, packets may have to be stored at intermediate nodes for transmission at a later time due to various reasons such as full buffers, packet loss, or scheduling. If an unlimited buffer space is available, the intermediate nodes need not have to reject or drop the arriving packets. However, in practice, buffers are limited in size. Although increasing the buffer space tends to minimize packet drops and increase the link utilization, large buffers have an adverse effect on the *latency*, *i.e.*, the delay experienced by packets stored in the network. Further, using larger buffer sizes at intermediate nodes would also result in secondary practical issues such as on-chip board space and increased memory-access latency. Consequently, a simple but fundamental question is the following: what is the minimum buffer requirement for each router given certain constraints on the throughput and queueing delay?

The problem of buffer sizing and congestion control is of paramount interest to router design engineers. Typical routers today route several tens of gigabits of data each second. Realistic studies have shown that, at times, Internet routers handle about ten thousand independent streams/flows of data packets. With a reasonable buffer size of few Gigabytes of data, each stream can only be allocated a few tens of data packets. Therefore, at times when long parallel flows congest a router, the effects of such a small buffer space per flow come to play. Though motivated by such practical concerns, our work is far from modeling realistic conditions. This work modestly aims at providing a theoretical framework to understand the fundamental limits of single information flow in finite-buffer networks and investigates the tradeoffs between throughput, packet delay and buffer size.

Broadly speaking, this work is in the area of performance modeling and analysis of networks. Our main objective is to develop a general framework for studying the latency and fundamental limits on the information carrying capacity of different networks (*e.g.*, wired/wireless, mobile/fixed topology) in a finite-buffer setting. We aim at completing the following major research tasks:

- Develop a theoretical framework to study the information-theoretic capacity of wired/wireless networks in finite-buffer regime.
- Study of the latency and throughput trade-offs with the buffer size in general wired/wireless networks.
- Extension to various communication, routing, source-traffic, scheduling scenarios.
- Modeling the dynamics of finite-buffer random linear network coding in general networks.

Next, we motivate our problem by explaining the necessity of addressing the performance analysis of networks in finite buffer regime, and present the previous works in the literature relevant to this topic.

1.1 New Trends in Buffer Size Reduction for Internet Routers

Until quite recently, Internet routers were widely believed to need large buffers. Typically, the size of router buffers is determined by the well-known “bandwidth-delay product” (BDP) rule-of-thumb [1]. The BDP states that the buffer size should be equal to the bandwidth of the link multiplied by the round-trip time (RTT) of a TCP[1] connection that can be bottlenecked at that link. Today, backbone links commonly carry around $10,000$ flows and operate at 2.5 Gb/s or 10 Gb/s. Hence, for a typical RTT of 250 ms and link bandwidth of 10 Gb/s, BDP mandates a buffer size of 2.5 Gb. Assuming 1000 bytes per packet, the approximate buffer storage is equal to $300K$ packets which is shared by all TCP

[1]Transmission control protocol

flows. Assuming only 500 concurrent flows [2], this is about 600 packets per flow, on the average. However, a recent work by Appenzeller [2], referred to as the “Stanford model”, has challenged the BDP rule advocating the use of much smaller buffers. The authors in [2] proposed an alternative rule $BDP/\sqrt{N}$ instead, where N is the number of flows, resulting in significantly smaller buffers (*i.e.*, only $3K$ packets vs $300K$ in the above example, which is shared by all the flows). Equivalently, this is about six packets per concurrent flow, on the average!

With declining memory prices, why not just over-buffer routers? The reasonings behind the Stanford model are explained in great detail in [2–4]. We briefly discuss two of them in the following. (1) Large buffers complicate the design of high speed routers, leading to high power consumption and more board space. If a few dozen packet buffers would suffice, then packet buffers could be incorporated inside the network processor (or ASIC) in a small on-chip SRAM; in fact, the buffers would only occupy a tiny portion of the chip. Hence, not only would external memories be removed, but also it would allow the use of fast on-chip SRAM or all-optical buffering, which scales in speeds that are much faster than DRAM. Additionally, with recent advances in all-optical switching, low storage all-optical buffering will open the door to routers with huge capacity and lower power than electronic routers. (2) Over-buffering increases end-to-end delay, conflicting with real-time applications such as online gaming, audio-video services, IPTV, and VoIP that have the UDP type traffic. Additionally, large buffers may increase the variance of the latency, making congestion control algorithms unstable. Since the work of [2], other researchers have also advocated small buffer sizes [5–17]. Some have studied new congestion control mechanisms for small buffer routers [18–21]. The experimental work is also performed to validate this direction [22, 23].

In summary, today, we have arrived at a juncture wherein the Internet handles a large volume of data. Realistic studies have shown that, at times, Internet routers handle about $10,000$ independent flows of data packets. Hence, only a few tens of data packets from each flow can be stored. As discussed above, the storage of buffers per flow, on the average, varies from six packets (using the Stanford model) to 600 packets (using the current BDP rule), assuming 500 concurrent flows. Therefore, at times when long parallel flows congest a router, the effects of such a small buffer space per flow come to play. Our work is motivated by such concerns. The matter gets worse for multi-hop wireless networks where the buffer sizes at the intermediate nodes are more restricted, due to wireless device and link constraints. Therefore, several fundamental questions rise. What is the capacity of finite-buffer wired/wireless networks and how does it vary with the buffer size? What is the latency of the finite-buffer wired/wireless networks and its variation with the buffer size? What is the interplay of the latency and throughput in a finite-buffer regime?

1.2 Finite-Buffer Networks vs. Information Theory

A finite-buffer queue has been studied in the information theory community as a finite-state Markov channel for a communication link in [24–26]. Various coding strategies for achieving capacity in infinite-buffer erasure line networks is outlined in [27]. Later, [28] considered the limitations posed by finite memory, specifically in a simple line network involving a single intermediate node. Inspired by this work, [29, 30] investigated bounds for the capacity of general multi-hop wireline networks. Several challenges arise when extending the study from a single intermediate node to a multi-hop line network as detailed in [31].

Advances in the area of error control coding have already led to the design of capacity-achieving codes for channels such as the binary erasure channel [32–39]. The design of such good codes has kindled greater interest in the study of theoretical limits such as capacity and throughput in several classes of wired and wireless networks. For example, for multicast in wireline networks, it was shown that the max-flow min-cut upper bound can be achieved [40–42] if every node sends out packets generated by random linear combinations of previously received packets. However, this technique assumes that nodes store all previously received packets, resulting in buffer growth as the source injects new packets in the network. Since then there has been considerable work in the areas of capacity study and network coding for both wired and wireless networks under the infinite-buffer assumption [27, 43–52]. Despite all the exciting results, the study of capacity of networks with finite buffer sizes has been limited. This can be attributed to the fact that analysis of finite-buffer systems are generally more challenging.

1.3 Finite-Buffer Networks vs. Queueing Theory

Studying capacity alone, information theory assumes infinite buffer; hurting the information delay which is of great interest for communication network community. Queuing theory, on the other hand, provides analysis for the delay with little regard to capacity. The problem of studying lossy networks with finite buffers has been investigated in the area of queueing theory [53–58]. The queueing theory framework attempts to model the packets of the network as customers, the delay due to packet loss over links as service times in the nodes, and the buffer size at intermediate nodes as the queue size. Also, the phenomenon of packet overflow in the network can be modeled by blocking (commonly known as *type II* or *blocking after service*) in queueing networks [59]. However, this packet-customer equivalence fails in general network topologies due to the following reasons. When the communication network contains multiple disjoint paths from the source to the destination, the source node can choose to duplicate packets on multiple paths to minimize the delay. This replicating strategy cannot be captured directly in the customer-server based queueing model. Moreover, communication schemes such as network coding in finite-buffer networks introduces redundant innovative packets, which cannot be studied using such a framework.

Although the other works provide some insight into the analysis of capacity of networks, they are limited to either infinite-buffer cases or finite-memory line networks with a simple single intermediate node. Moreover, the interplay of the throughput and latency with the buffer size is not considered. All of this inspires us to obtain a framework to study the effect of finite memory in general multi-hop wired/wireless networks. In the process, we expect to develop tools and techniques that will be suitable for analyzing the capacity, and the throughput-latency trade-off in such networks in various communication scenarios such as unicast, multiple unicast, and multicast. Clearly, the previous techniques do not lead to the goals that are sought out in our proposed research.

1.4 Finite-Buffer Networks with Scheduling Policies

Since the seminal paper of Tassiulas and Ephremides which proposed a throughput-optimal joint routing/scheduling algorithm [60] (backpresure routing), there has been a great effort to develop throughput-optimal schemes for different networks [61–66]. [61, 62] investigated throughput optimal scheduling policies for finite-buffer wired and wireless networks with performance guarantees. Also recently, [67] has proposed a buffer management strategy to improve the delay-throughput trade-offs in backpressure routing. Network coding in wireless queueing networks has also been studied to examine the effects of the saturated queues [68, 69]. However, both [68] and [69] considered the infinite queue over single-hop channels and examined the stability condition to ensure the delay will not grow without bound. Recently, [70] presents a rough estimate for the performance of finite-buffer networks. However, as explained before, the tools developed in queueing theory are inadequate for the analysis of random linear coding scheme, which we employ to study the capacity of general finite-buffer networks. In Chapter 5, inspired by the routing/scheduling scheme used in [63] for wireless erasure networks and in [62] for the finite-buffer case, we adapt a modified backpressure routing policy for the sake of our analysis.

CHAPTER 2

TOWARDS A GENERAL FRAMEWORK FOR ANALYSIS OF FINITE-BUFFER NETWORKS

In this chapter, we first motivate our study of finite-buffer networks by some examples. Then, we introduce the notations and definitions used in our proposed framework for analysis of finite-buffer networks. Finally, we present the general framework and its application in performance analysis and modeling.

2.1 Preliminary Discussion and Motivation

There are three major observations that we wish to present to motivate our study of finite-buffer networks. First, for small-size buffers as in the Stanford model, the max-flow min-cut result (of the infinite-buffer case) does not hold and, hence, a new framework and tool must be developed to obtain maximum achievable rates. Second, to our knowledge, the relation of latency with the buffer size in the finite-buffer regime remains unknown in most cases. Additionally, there is a trade-off between delay and capacity (or throughput in general). In particular, the penalty, in the form of increased latency, is severe for any subtle improvement in the information rate. Third, the type of the node (defined below) is a determining factor for allocating the buffer space to various incoming flows (i.e., the buffer management strategy). Our study is fruitful for network engineering by shedding light on the above issues. Here, we present some simulation results to establish these motivating factors and we defer some details to the later subsections. We will consider a discrete-time model in which every node transmits one packet per epoch to the next-hop node. First, an eight-hop line network with the probability of packet erasure on each link set to $\varepsilon = 0.2$ is considered. Note that a finely discretized version of the system approximates the dynamics of the continuous-time system to any degree of precision for an appropriate choice of link erasures in the discrete model [31]. That explains the use of relatively large link erasures.

At each setup of experiment, for all intermediate nodes, the same buffer sizes m of 10, 25 or 500 packets are used (which covers the range from the Stanford model to the BDP model, refer to the introduction chapter). It is assumed that the source has infinite number of information packets and it attempts to send one packet to the next-hop neighbor in every epoch with probability R_s. We denote R_s as the source injection rate. Figure 2.1 shows the trade-off between the throughput and average information packet delay for three different buffer sizes obtained from the actual simulations. It is concluded that both throughput and

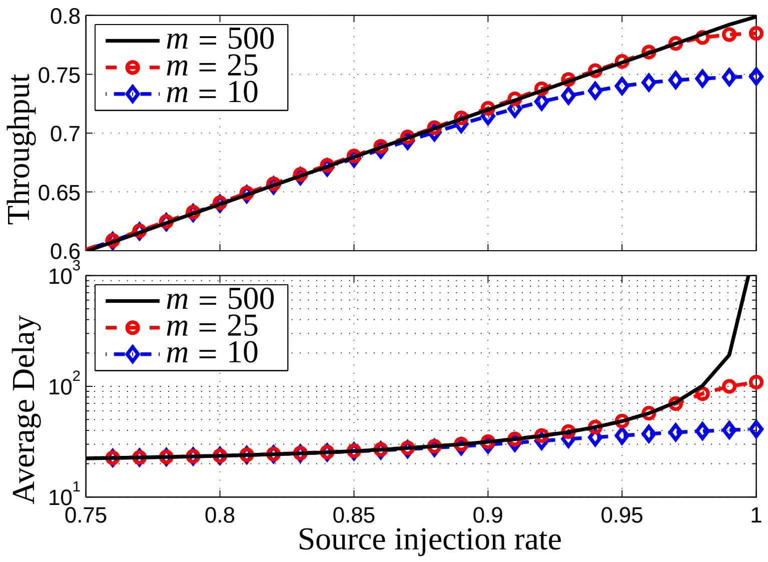

Figure 2.1: Throughput and delay versus the source injection rate.

delay increase with the buffer size. However, the delay undergoes an exponential growth compared to the throughput which saturates to the min-cut value of the line. In theory, the min-cut value ($1 - \varepsilon_{max} = 0.8$) is achieved when the buffer size approaches infinity for a very large block size. We also observe a large loss in the throughput if the average delay is bounded. For example, when the average delay is 26 epochs (i.e., $R_s = 0.85$), the throughput gap from the min-cut value is 16% for the considered buffer sizes.

Next, we consider two cases of a four-hop line network (with the vertex set $\{s, v_1, v_2, v_3, d\}$, where s and d are the source and the destination nodes, respectively) with packet erasure probability for each hop from the set $\mathcal{E} = \{0.3, 0.5, 0.5, 0.2\}$. Further, intermediate nodes

v_1, v_2, and v_3 have the same buffer of 10 packets (or 20 packets for the second setup) each. We assumed a lossless hop-by-hop feedback scheme (without any coding). Now, one might ask the question: *when it comes to buffer management strategy, is the more, the merrier?* Surprisingly, sometimes it is damaging to use all buffer slots for the same flow even when the space is available. To illustrate this claim, the estimated distribution of packet occupancy at the intermediate nodes under both buffer sizes is presented in Figure 2.2. As the buffer size is doubled, it is noticed that node v_1 that is congested remains congested, node v_3 that has low occupancy registers a marginal change in the occupancy distribution, and node v_2, which is the node following the bottleneck edge in the network, registers a significant change in the occupancy distribution.

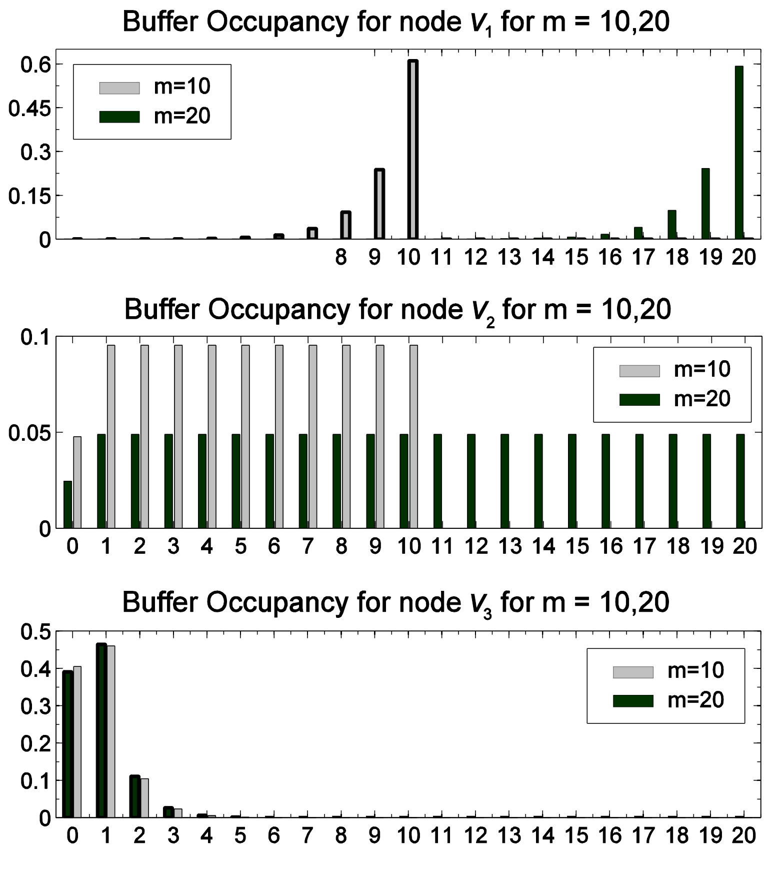

Figure 2.2: Estimated buffer occupancy distribution for different types of nodes.

Figure 2.3 presents the simulation for the contribution to capacity and average delay by varying the buffer size of one node while keeping the buffer size of the remaining two nodes fixed at 10 packets. Using Figure 2.3, we conclude that contribution to mean delay from v_1 increases almost linearly with the increase in its buffer size whereas the change in capacity is subtle. Changing the buffer-size of v_3 is insignificant on both capacity and delay. Lastly, doubling the buffer size of v_2 increases both the capacity and average packet delay. However, the rate of increase of the delay with the increase in the buffer size of v_2 is smaller than that caused by the increase of the buffer size of v_1. By identifying these

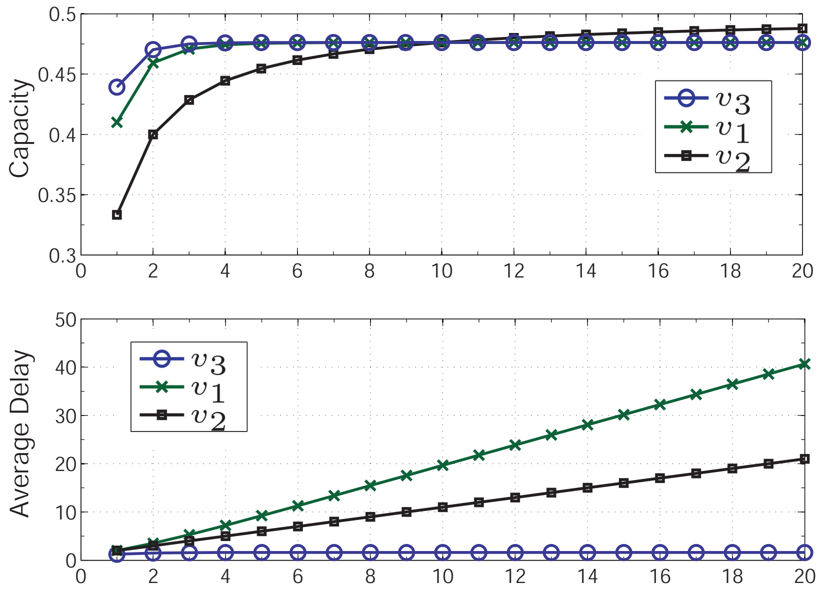

Figure 2.3: Capacity and average delay contribution of each node versus buffer size.

nodes, our proposed work will make it possible to design buffer management strategies that make efficient use of the buffers without compromising much on the throughput or average packet delay. For example, if the capacity of the four-hop network example above is to be maximized albeit with a reasonable price in the average delay, all 20 buffer slots of v_2 must be used for the flow whereas that of v_1 may be kept at around four. In the following subsections, we first present the framework for finite-buffer analysis, irrespective of the underlying communication scenario or the loss recovery scheme. Then, we present the specific application of the framework for a line network with erasure links.

2.2 Models and Definitions

The following notations and definitions will be used in the next chapters. However, at each chapter, we will try to clarify the notations and definitions that are not precisely defined here, or even may reiterate them to avoid confusion.

We consider each communication channel to be a memoryless erasure channel. We model a "general" packet erasure network as an acyclic directed graph $\overrightarrow{G}(V, \overrightarrow{E})$. The elements of V are called *nodes* and the directed pairs of V that are elements of $\overrightarrow{E}$ are called *links*. A node u can communicate with another node v if and only if there is a link between them, i.e., $(u, v) \in \overrightarrow{E}$. The links are assumed to be unidirectional, memoryless and lossy, i.e., packets transmitted on a link $\overrightarrow{e} = (u, v) \in \overrightarrow{E}$ are lost at random with a probability of $\varepsilon_{\overrightarrow{e}} = \varepsilon_{(u,v)}$. Note that the erasures are due to the quality of links and do not represent packet losses due to finite buffers. We assume that every intermediate node u is equipped with a fixed buffer size m_u packets per flow. This implies that the entire relaying storage of every node is divided to segments of size m_u packets, one segment for each flow. We assume that communication happens in a discrete-time fashion. In each unit of time, referred as *epoch*, a packet per flow is transmitted by a node on each outgoing link. It is assumed that the source and the destination nodes has no buffer constraints. The unicast information theoretic capacity between the source and destination in a network is defined to be the maximum achievable rate of transmission of information packets (in packets per epoch) between the pair of nodes[1]. Throughout this work, we distinguish between information theoretic capacity (in short capacity) and the throughput of a scheme. Here, the throughput of a scheme is defined to be the rate of transmission of information packets (in packets per epoch) between the source and destination nodes for a fixed communication scheme. Further, unless a source packet arrival model is defined, we assume that the source node can generates new information packets at every epoch. The delay or latency of a packet

[1]The maximum is calculated over all possible means used for packet generation and buffer update at network nodes.

is defined as the time taken from the instant when the source starts serving the packet by injecting it to the network to the instant when the destination node obtains it.

Further, we employ the following notations. Let $\mathcal{N}^{-}(u)$ denote the set of all the neighboring nodes that u can send packets via outgoing links. Likewise, the set $\mathcal{N}^{+}(u)$ is defined using the incoming links of u. For any $x \in [0, 1]$, $\overline{x} \triangleq 1 - x$. Node s and node d represent the source and the destination nodes respectively.

2.3 Analytical Framework for Finite-Buffer Analysis

In this section, our objective is to develop a general analytic framework which can be applied to as many communication scenarios and network coding/routing schemes as possible. The first step in the construction of the proposed framework is the characterization of the buffer states of the intermediate nodes at each particular epoch. Often, this characterization is simple because the occupancy of a node is measured by the number of packets presently stored at the node. However, there are exceptions to such characterization, and it occurs when the occupancy of each buffer cannot be described by the number of physical packets stored. For example, when using random linear coding (RLC), even with a single packet reception, the entire buffer of a node becomes physically full with multiples of the same packet. Thus, although the node's buffer would always be physically full, its occupancy is measured as the number of packets of information (*i.e.,* innovative or linearly independent encoded packets) stored. In Chapter 3, we will see that for a line network scenario, the characterization of buffer states with RLC is exactly the same as a non-RLC scheme such as hop-by-hop feedback. However, in Chapter 9, complications of modeling the buffer states for RLC in a general network topology will be described.

For any single-copy routing scheme, we can assume the number of packets physically stored at each node as the buffer state. In such routing schemes, a node forwards each packet to only one other node and deletes the packet from its buffer which leads to increasing the occupancy of the receiving node and decreasing the occupancy of the sending node, both by one. Hence, the state of the network buffers can be clearly defined. Specifically, the occupancy of node $u \in V$ is denoted by the state of a queue $X_u(t)$. We say the queue $X_u(t)$ is full and hence would block an incoming packet (to node u) if its state is equal to m_u (the size of the buffer). Obviously, the incoming packet would not be blocked only if the state of the queue is less than m_u.

The above setup provides an elegant way of analyzing the performance of the system using *Markov chains* (MC). We proceed in modeling the problem using a discrete-time finite-state MC. For every node u, consider a queue $X_u(t)$. For a network of N intermediate nodes, we need N queue variables. Further, each queue variable can take values from 0 up to m_u. Assuming that all the nodes have a buffer size of m packets, we see that the number of states in the MC that is needed to completely track the dynamics of the network is in the order of $(m+1)^N$; growing exponentially with N. Note that transitions between states in the chain is based on the channel realization at that time instant. For instance, it must be noted that *successful transmission* ("*conveyance*") of a packet from u to a next-hop neighbor v is possible only when the packet is not erased by the channel and when node v's queues are not full. Upon a successful transmission on an edge from a node u to v, both the queues corresponding to the two nodes must be updated.

Due to the exponential growth in the size of the exact Markov chain in most of the scenarios, exact calculation of the steady-state probabilities and the network performance is computationally intractable. Further, the finite-buffer constraint introduces a strong dependency in the state transitions of a queue at node u on the state of its next-hop neighbors. Finally, the intractability of the EMC is compounded by a non-memoryless output process at each node. Thus, approximation is a more favorable option.

We propose an approximation method that updates a queue for node u considering: (1) The effect of blocking imposed by the next-hop neighbors $\mathcal{N}^-(u)$ (on the packets departing from u), and (2) The packet arrival process at u from the previous-hop neighbors $\mathcal{N}^+(u)$. Hence, we will only consider the dependency of the state transition probabilities of the queue for each node u to the state of the queues corresponding to nodes in $\mathcal{N}^{+}(u)$ and $\mathcal{N}^{-}(u)$. Note that this will be exact for most of the schemes on a general network as well as when RLC is applied on a line network. The main idea of the approximation framework is to divide the multi-dimensional MC with multiple reflections into multiple simple MCs whose steady-state probabilities can be calculated independently. Note that although each MC process is assumed independent of the other MC processes, the interdependency of the states of their queues are captured by the approximation method via their steady-state probability distributions.

Let $\mathcal{X}$ denote the set of all the queues $X_u(t)$ required for the analysis, where $u \in V$. Note that, each queue $X_u(t) \in \mathcal{X}$ must form an irreducible ergodic Markov chain whose state transition probabilities must be systematically computable given all the information regarding the communication scheme, buffer management strategy, the erasure probabilities on the links, and the network topology. As a result, all the MCs will have unique steady-state probability distributions which are denoted by $\pi_u(\cdot)$, *i.e.*, $\pi_u(k) = Pr\{X_u(t) = k\}$ for $k = 0, 1, \ldots, m_u$. It is notable that in general, by means of our approximation method, the state transition probabilities for MC of every queue $X_u(t)$ depend on the steady-state distributions of the queues corresponding to nodes in $\mathcal{N}^{+}(u)$ and $\mathcal{N}^{-}(u)$. Since there is no prior information about the probability distribution of these queues, the proposed estimation must be done iteratively. To determine the state transition probabilities for each MC, we need to know the dynamics of arrival and departure of packets to its corresponding queue. In general, for every queue $X_u(t) \in \mathcal{X}$, we define multiple incoming and outgoing streams of packets which are assumed to be statistically independent for the purpose of our approximation procedure. Let $\Lambda_u = \{\lambda_1, \ldots, \lambda_{z_{in}}\}$ be the set of arrival rates, where z_{in} is the number of arriving streams at the queue $X_u(t)$. Similarly, let $\Omega_u = \{\mu_1, \ldots, \mu_{z_{out}}\}$ be the set of departure rates, where z_{out} is the number of departing streams. Thus, at each epoch, the total number of arriving packets can range from 0 to z_{in}, since each arrival occurs with probability λ_i for $i = 1, \ldots, z_{in}$. Similarly, the number of departing packets can range from

0 to z_{out}. Hence, at each epoch, given $X_u(t) = n_u$, its state can change to any other state in the set $\{\max\{n_u - z_{out}, 0\}, \ldots, \min\{n_u + z_{in}, m_u\}\}$. The state dynamics of the queue $X_u(t)$ at the t^{th} epoch is a Markov chain that is similar to the one depicted in Figure 2.4. Given the current state of the queue, *i.e.*, $X_u(t) = n_u$, we define its arrival and departure polynomials as

$$A^{(n_u)}(x) = \sum_{k=0}^{z_{in}} a_k^{(n_u)} x^k = \prod_{j=1}^{z_{in}} (\overline{\lambda_j(n_u)} + \lambda_j(n_u)x) \quad , \qquad E^{(n_u)}(x) = \sum_{k=0}^{z_{out}} e_k^{(n_u)} x^k = \prod_{j=1}^{z_{out}} (\overline{\mu_j(n_u)} + \mu_j(n_u)x),$$

where $a_k^{(n_u)}$ can be interpreted as the probability of the event that the number of packet arrivals to the queue $X_u(t)$ is equal to k in a single epoch. Similarly, $e_k^{(n_u)}$ can be interpreted as the probability of the event that the number of packet departures from the queue $X_u(t)$ is equal to k in a single epoch. The superscript on the coefficients represents the current state of the queue, $X_u(t) = n_u$. We included this dependency of the arrival and departure polynomials on the current state of queue to account for some cases such as the wireless networks with backpressure routing as we will see in Chapter 5.

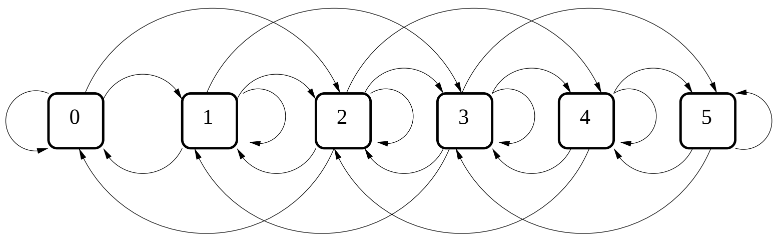

Figure 2.4: MC of the queue $X_u(t)$ with $m_u = 5$ and $z_{in} = z_{out} = 2$.

To put everything in a more systematic form, let $\Delta_u = \{A^{(n_u)}(x)\}_{n_u=0}^{m_u}$ and $\Gamma_u = \{E^{(n_u)}(x)\}_{n_u=0}^{m_u}$ be the sets of arrival and departure polynomials for the queue $X_u(t)$, respectively, where m_u is the buffer size of node u. Given Δ_u and Γ_u, the queue's state transition probabilities can be easily computed. As an example, for $0 < j < m_u$, we have the following:

$$Pr\{X_u(t+1) = j | X_u(t) = i\} = \sum_{k=0}^{z_{in}} a_k^{(i)} e_{k+i-j}^{(i)} \ . \tag{2.1}$$

For notational consistency, we can extend $e_k = 0$ for $k < 0$ or $k > z_{out}$ and $a_k = 0$ for $k < 0$ or $k > z_{in}$. As a result, the proper approximate MC is formed for $X_u(t)$ with steady-state probability distribution $\pi_u(\cdot)$.

In summary, given all the information regarding the performance problem, for any node u, a queue with its incoming and outgoing streams will be identified properly. Then, the corresponding arrival and departure polynomials will be obtained in a parametrical fashion. These polynomials describe the state transitions of the queue from which the steady-state probability distribution can be computed for the MC. Then, we propose the following algorithm, denoted as the "iterative estimation algorithm" (IEA), to compute steady-state probability distributions for all queues:

Step 1. Initialization (iteration 0): Start with arbitrary values for $\Lambda_u^{(0)}$ and $\Omega_u^{(0)}$ and compute $\Delta_u^{(0)}$ and $\Gamma_u^{(0)}$ for each queue, where the superscript denotes the iteration number. However, apply prior information regarding the queues for initialization. For example, in our initial model, destination node does not block any arriving packet. Also, the source node has infinitely many packets.

Step 2. Increase the iteration index by one (e.g., iteration i). Given $\Delta_u^{(i-1)}$ and $\Gamma_u^{(i-1)}$ for all the queues $X_u(t) \in \mathcal{X}$, compute their steady-state probability distributions $\pi_u^{(i-1)}(\cdot)$.

Step 3. Given $\pi_u^{(i-1)}(\cdot)$ for all the MCs, compute the new sets of arrival/departure polynomials $\Delta_u^{(i)}$ and $\Gamma_u^{(i)}$ for each queue $X_u(t) \in \mathcal{X}$.

Step 4. Go back to Step 2 until all the steady-state probabilities converge to fixed distributions.

Note that at Step 3, using $\pi_u^{(i-1)}(\cdot)$ for all queues, we compute each arrival rate λ_i by applying its definition for each queue $X_u(t) \in \mathcal{X}$. That is, λ_i is the probability that a packet arrives on the stream i (without being erased), which may or may not be blocked by u. Hence, λ_i is computed by multiplying the probability of two events: 1. The event that the packet is not erased by the corresponding incoming link, and 2. The event that the corresponding queue of the node in $\mathcal{N}^+(u)$ is not in the empty state (obtained at Step 2 of IEA). Similarly, for each queue $X_u(t) \in \mathcal{X}$, we compute the departure rate μ_j by using its definition, which is the probability of the event that the link j does not erase the packet, and the event that the queue of the corresponding receiving node in $\mathcal{N}^-(u)$ is not at the full buffer state (and hence, it does not block the packet from u). The calculation of Δ_u and Γ_u from Λ_u and Ω_u will be straightforward then. Finally, once steady-state distributions of all queues are computed, we can obtain analytical expressions for the performance parameters such as capacity, throughput, and latency distribution. This will be discussed and in more details in the following Chapters.

CHAPTER 3

ANALYSIS OF THROUGHPUT AND DELAY IN LINE NETWORKS

In this chapter, we study the effects of finite buffers on the throughput and delay of line networks with erasure links.[1] As identified in Chapter 2, the calculation of performance parameters such as throughput and delay is equivalent to determining the stationary distribution of an irreducible Markov chain. We note that the number of states in the Markov chain grows exponentially in the size of the buffers with the exponent scaling linearly with the number of hops in a line network. We apply the proposed iterative estimation algorithm to approximately identify the steady-state distribution of the exact Markov chain by decoupling the chain into smaller chains. The approximate solution is then used to analytically characterize the effect of buffer size on throughput and distribution of packet delay. Further, the results of this chapter can be used to classify nodes based on congestion that yields an intelligent scheme for memory allocation using the proposed framework. An example of such applications is presented in Chapter 2.1. Finally, simulations will confirm that our framework yields an accurate prediction of the variation of the throughput and delay distribution.

As mentioned in Chapter 1.1, in [28], Lun *et al.* consider a discrete-time model, where each node can transmit and receive a packet during each epoch, to analyze the capacity of a simple two-hop lossy network. In [29], upper and lower bounds on the throughput of line networks are derived, but were unable to provide good approximations for packet delay and buffer occupancy statistics. While our approach employs a model of network similar to that in [28, 29], we extend their results not only to derive estimates for the throughput of line networks of any hop-length and intermediate node buffer size, but also to derive quantitative estimates for packet delay distribution.

[1]This work is done in collaboration with my former lab-mate, Dr. Badri N. Vellambi [31].

3.1 Problem Statement and Network Model

We define a line network of hop-length h to be a directed graph with vertex set $V = \{s = v_0, v_1, v_2, ..., v_{h-1}, d = v_h\}$ and edge set $\overrightarrow{e} = \{\{v_i, v_{i+1}\} : i = 0, ..., h-1\}$. The links are assumed to be unidirectional, memoryless and lossy with erasure probability ε_i on link $\{v_{i-1}, v_i\}$ for $i = 1, ..., h$. A lossless hop-by-hop acknowledgement setup is in place to indicate the successful receipt of a packet[2]. Moreover, the packet processes on different links are assumed to be independent. Each node $v_i \in V$ has a buffer of size m_i packets with each packet having a fixed size. Note that the buffer size can vary with the node index. Lastly, the source and destination nodes are assumed to have sufficient memory to store any amount of data.

The system is analyzed using a discrete-time model, where each node can transmit at most one packet over a link per epoch. The unicast capacity between a pair of nodes is defined to be the supremum of all achievable rates of transmission of information packets (in packets per epoch) between a pair of nodes. The supremum is calculated over all possible means used for packet generation and buffer update at intermediate nodes. Note that the source node can generate innovative packets during each epoch. For instance, in the particular case of the line network defined above, we would like to identify the unicast capacity between the source v_0 and the destination v_h.

Before we proceed to the modeling, we briefly motivate the assumed discrete-time model with an example. Consider a continuous-time model with the discrete-time model for varying times of epoch for a simple continuous-time two-hop line network with a Poisson packet generation process at the source with parameter $\lambda_1 = 10$ pkts/sec. The service time at the intermediate node is also Poisson with parameter $\lambda_2 = 10$ pkts/sec, and the links connecting the source to the intermediate node and the intermediate node to the destination are both packet-erasure channels with erasure probabilities $\varepsilon_1 = \varepsilon_2 = 0.1$. Finally, suppose that the intermediate node has a finite buffer of $m = 10$ packets. Figure 3.1 presents the

[2]This assumption is made to simplify modeling. In the absence of perfect ACK, one can use random linear coding over a large finite field to achieve the same desired throughput. See Section $III - D$ in [31].

(simulated) capacity for the continuous model and the time-discretized models for various epoch durations. It is noticed that as the epoch duration is made smaller, the discrete-time model becomes more accurate in predicting the capacity. This was verified to be the case for all line networks with Poisson arrivals and service times.

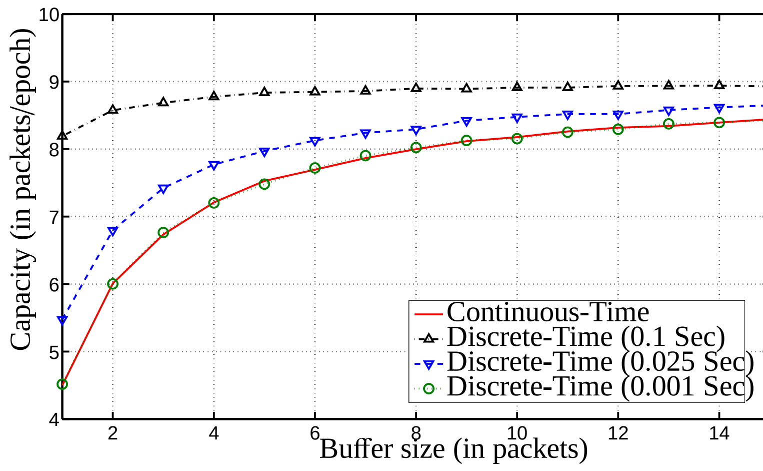

Figure 3.1: An illustration of the precision of the discrete-time model.

Lastly, we use the following notations. $\mathbb{G}(p)$ denotes the geometric distribution with mean inter-arrival time $\frac{1}{1-p}$. $\sigma(\cdot)$ denotes the indicator function for $\mathbb{Z}_{>0}$. For any $x \in \mathbb{R}$, $\overline{x} \triangleq 1 - x$. Finally, $\otimes$ denotes the convolution operator.

3.2 Finite-Buffer Analysis

In this section, we will apply our general framework of finite-buffer analysis to the problem of identifying buffer occupancy distributions, and consequently, performance parameters such as throughput in line networks.

One of the most important performance parameters of a network is its throughput and the problem of identifying capacity is directly related to the problem of finding schemes that are *rate-optimal*. In our model of line network, a scheme that performs the following in the same order can be seen to be rate-optimal.

1. If the buffer of a node is not empty at a particular epoch, then it must transmit at least

one of the packets.

2. A node deletes the packet transmitted at an epoch if it receives an acknowledgement from the next hop.

3. A node accepts an arriving packet if it has space in its buffer. It then sends an ACK to the previous node.

In the absence of feedback, rate-optimality can be achieved by employing random linear combinations based network coding over a large finite field as is described in [28, 29].

In order to model the network with lossless feedback, we need to track the number of packets that each node possesses at every instant of time. We do so by using the rules of buffer update under the optimal scheme. Let $\mathcal{X}(t) = (X_1(t), \ldots, X_{h-1}(t))$ be the vector whose i^{th} component denotes the number of packets the i^{th} intermediate node possesses at time t. Also, let $\mathbf{E}(t) = (E_1(t), \ldots, E_h(t))$ be a vector of channel conditions at time t, where $E_i(t) = 1$ if and only if the link (v_{i-1}, v_i) does not erase the packet at the t^{th} epoch.

Hence, we see that $\{\mathcal{X}(t)\}_{t\in\mathbb{Z}_{\geq 0}}$ forms a Markov chain. It is readily checked that this chain has $\prod_{i=1}^{h-1}(m_i + 1)$ states. Further, this chain is *irreducible, aperiodic, positive-recurrent*, and *ergodic* [71] and therefore has a unique steady-state probability. By ergodicity, we can obtain temporal averages by statistical averages. We then see that the computation of throughput is equivalent to the computation of the likelihood of the event that $X_h > 0$ and $E_h = 1$.

The exponential growth in the size of the chain and the presence of boundaries (due to finite buffers), exact calculation of the steady-state probabilities (and hence the throughput) becomes very cumbersome even for networks of reasonable buffer sizes and hop-lengths. The exact chain for the dynamics of the system is such that a state update at a node has a strong dependence on the states of both its previous-hop and its next-hop neighbors. Additionally, the process of packet transmission over intermediate edge can be shown to be non-memoryless. These facts add to the intractability of the exact computation of the distribution. However, it is possible to decouple the chain into several Markov chains with a single finite-boundary under some simplifying assumptions. To have an approximate decoupled model, we need to identify the transition probabilities of the decoupled chains, which is possible only if we know the arrival and departure processes on each edge. The rate of information on any edge is directly related to the fraction of time the sending node is non-empty and the fraction of time a successfully delivered packet will get blocked (and this happens if the receiving node is full at the time of packet arrival). Hence, to have a model for a node, we need to have the approximate buffer occupancy distributions for neighboring nodes. This hints naturally at an *iterative* approach to the problem. In this section, we develop an iterative estimation method that considers the effect of blocking with some simplifying assumptions. To develop an iterative technique, we assume the following.

A1. The packets are ejected from nodes in a memoryless fashion. Equivalently, we assume that $\Pr[(X_{i-1}(t) > 0) \wedge (E_i(t) = 1) | X_i(t) = k]$ does not vary with the occupancy k of the i^{th} node. This allows us to track just the information rate and not the exact statistics.

A2. The blocking event occurs independent of the state of a node, *i.e.*, $\Pr[(X_{i+1}(t) = 0) \wedge (E_{i+1}(t) = 1) | (X_i(t) = k)]$ is the same for $k = 1, \ldots, m_i$. This allows us to track just the blocking probability and not the joint statistics.

A3. At any epoch, given the occupancy of a particular node, the arrival process is independent of the blocking process.

Under these assumptions, for node v_i, the arrival stream of packets is coming only from the link (v_{i-1}, v_i) (*i.e.*, $z_{in} = 1$) with rate λ_i. Similarly, there is only one departure stream of innovative packets from node v_i, leaving through the link (v_i, v_{i+1}) (*i.e.*, $z_{out} = 1$) with rate μ_i. Here, the subscripts of arrival and departure rates denote the index of the node. Thus, given $z_{in} = z_{out} = 1$ and the arrival and departure rates, the set of arrival polynomials Δ_{v_i}

and departure polynomials Γ_{v_i} for node v_i (i.e., the queue X_i) can be simply obtained as

$$\Delta_{v_i} = a_0 + a_1 x = \overline{\lambda_i} + \lambda_i x, \qquad \Gamma_{v_i} = e_0 + e_1 x = \overline{\mu_i} + \mu_i x. \tag{3.1}$$

Then, we can show that the resulting MC[3] for node v_i with the buffer size m_i is given by the chain depicted in Figure 3.2, with the parameters obtained via (2.1) as $\alpha = a_1 e_0 = \lambda_i \overline{\mu_i}$, $\beta = a_0 e_1 = \overline{\lambda_i}\mu_i$, and $\alpha_0 = a_1 = \lambda_i$. Then, the steady-state distribution for the chain in Figure 3.2 can be computed (see Step 2 of the IEA algorithm in Chapter 2.3) using

$$\Pr\{X_i = k\} = \begin{cases} \frac{1}{1+\frac{\alpha_0}{\beta}\left(\sum_{l=0}^{m_i-1}\frac{\alpha^l}{\beta^l}\right)} & k = 0 \\ \frac{\frac{\alpha_0 \alpha^{k-1}}{\beta^k}}{1+\frac{\alpha_0}{\beta}\left(\sum_{l=0}^{m_i-1}\frac{\alpha^l}{\beta^l}\right)} & 0 < k \le m_i \end{cases}. \tag{3.2}$$

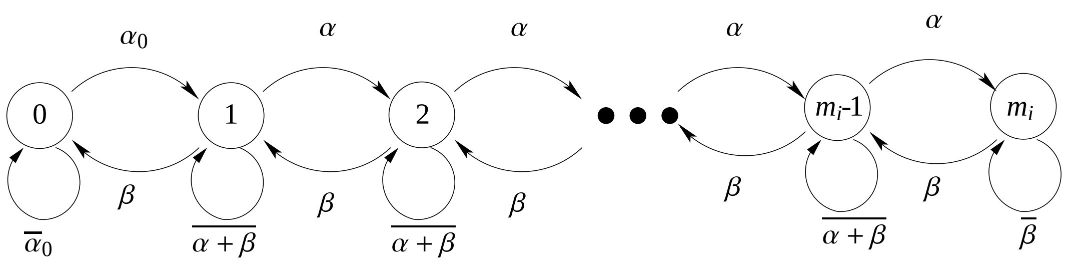

Figure 3.2: The chain for the node v_i obtained by the assumptions A1-A3.

The blocking probability that the node v_{i-1} perceives from the node v_i, assuming that v_i sees a blocking probability of $p_{b_{i+1}}$ caused by v_{i+1}[4], can then be calculated for $i = 1, 2, \ldots, h-1$ as follows.

$$p_{b_i} = (\varepsilon_{i+1} + \overline{\varepsilon_{i+1}} p_{b_{i+1}}) \Pr\{X_i = m_i\} \tag{3.3}$$

Then, we have $\mu_i = \overline{p_{b_{i+1}}}\,\overline{\varepsilon_{i+1}}$ for $i = 1, 2, \ldots, h-1$. Further, a packet arrives at node v_i only if it is not erased on the link (v_{i-1}, v_i) and buffer of node v_{i-1} is non-empty. Hence, for the arrival rate to node v_i, we have $\lambda_i = \overline{\varepsilon_i}\,(1 - \Pr\{X_{i-1} = 0\})$.

[3]Due to the discrete-time nature of the framework, two distinct MCs are associated with each intermediate node. Here, we considered the transmit first MC in which, at each epoch, the event of transmitting a packet occurs before the event of receiving a packet.

[4]Note that the arrival rate at the node v_1 is $\lambda_1 = \overline{\varepsilon}_1$ and that the blocking probability of v_h is zero,*i.e.* $p_{b_h} = 0$.

Given (3.1) and (3.2), the The approximate solution to the buffer occupancy distributions for each intermediate node follows from our proposed iterative estimation algorithm introduced in Chapter 2.3. Finally, the estimate of the throughput capacity can be obtained from the approximate solution as

$$C = \lambda_h^*,$$

where λ_h^* is the approximate packet arrival rate (result of the convergence after iteration) at the destination.

3.3 Packet Delay Distribution

In this section, we use the approximate solution of Section 3.2 to obtain the estimates on the probability distribution of the delay of a packet. We define the packet delay as the time taken from the instant when the source starts sending the packet to the instant when the destination receives it. We assume a *first-come first-serve* treatment of packets at the intermediate node buffers.

In order to compute the distribution of delay that a packet experiences in the network, one can proceed in a hop-by-hop fashion. Considering the last relay node, the additional delay of an arriving packet (at time t) at node v_{h-1} depends on the occupancy of the node v_{h-1} and the erasure channel that follows it to the destination. Suppose at epoch t, node v_{h-1} has $k \le m_{h-1} - 1$ packets in addition to the arriving packet. Then, the packet has to wait for the first k packets to leave before it can be served. Since each transmission takes place independently, the distribution of delay is sum of $k + 1$ independent geometric distribution with mean inter-arrival time $\frac{1}{1-\varepsilon_h}$, which is denoted by $\otimes^{k+1}\mathbb{G}(\varepsilon_h)$. Suppose that the distribution of buffer occupancy *at time of packet arrival* is given by $\pi_{h-1}(i)$, then the distribution of delay added by v_{h-1} to the packet is

$$\mathbf{D}_{h-1} = \sum_{i=0}^{m_{h-1}-1} \pi_{h-1}(i) \otimes^{i+1} \mathbb{G}(\acute{\varepsilon}_h). \tag{3.4}$$

However, the situation is different for other intermediate delays because of the effect of blocking. The additional delay incurred while being stored at the node v_j, $0 < j < h - 1$, is given by

$$\mathbf{D}_j = \sum_{i=0}^{m_j-1} \pi_j(i) \otimes^{i+1} \mathbb{G}(\varepsilon'_{j+1}), \tag{3.5}$$

where we used the following to consider blocking.

$$\varepsilon'_i = \begin{cases} \varepsilon_i + \theta_{v_i}(m_i)(1-\varepsilon_i) & i = 1, 2, \ldots, h-1 \\ \varepsilon_h & i = h \end{cases}, \tag{3.6}$$

where $\theta_{v_i}(k)$ is the steady state probability of that node v_i already has k packets when the packet is transmitted successfully from v_{i-1}. $\pi_j(i)$ and $\theta_{v_i}(k)$ are related by

$$\pi_j(i) = \begin{cases} \frac{\theta_{v_j}(i)}{1-\theta_{v_j}(m_j)} & i = 1, 2, \ldots, m_j - 1 \\ 0 & i = m_j \end{cases}. \tag{3.7}$$

By assuming that the delays incurred by each node and its adjoining outgoing link is independent of each other, we obtain the total delay considering all hops to be

$$\mathbf{D} = \mathbb{G}(\varepsilon'_1) \otimes \mathbf{D}_1 \otimes \cdots \otimes \mathbf{D}_{h-1}.$$

Hence, the delay distribution is known if the steady-state distributions of buffer states $(\pi_j(\cdot),\ \ j = 1, ..., h-1)$ as seen by arriving packets is known. However, it is a simple exercise to derive these distributions from the results of Section 3.2. The method of deriving both the transmit-first and receive-first distributions are described in details in Chapter 4.

3.4 Simulation Results

We have so far presented some fundamental tools for finite-buffer analysis of line networks. In this section, we show that they are very helpful to obtain accurate estimates of the performance parameters such as throughput, delay distribution and buffer occupancy distribution for line networks.

To understand the variation of our throughput capacity estimate of Section 3.2, in each of the figures, the simulation of the actual capacity is presented in addition to our analytical results. Figure 3.3 presents the variation of the capacity with the hop length for a network with each intermediate node having a buffer size of five packets. Moreover, the simulations are performed when the probability of erasure on every link is set to either 0.25 or 0.5. It is noticed that the estimate captures the variation of the actual capacity of the network within about 1.5% of error.

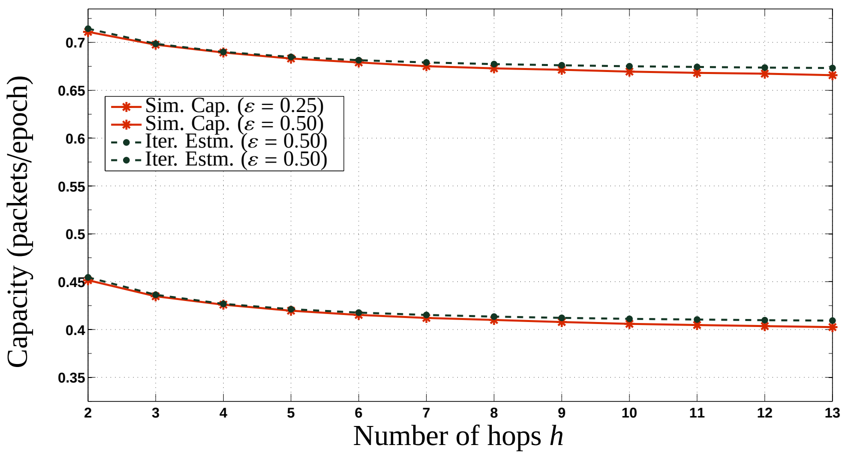

Figure 3.3: Capacity of a line network with $m = 5$ vs. the number of hops h.

In order to study the effect of buffer size, we simulated a line network of eight hops having the same erasures as in the previous setting. Figure 3.4 presents the variation of our results and the actual capacity as the buffer size of the intermediate node is varied. It can be seen that as the buffer size is increased, all curves approach the ideal min-cut capacity of $1 - \varepsilon$.

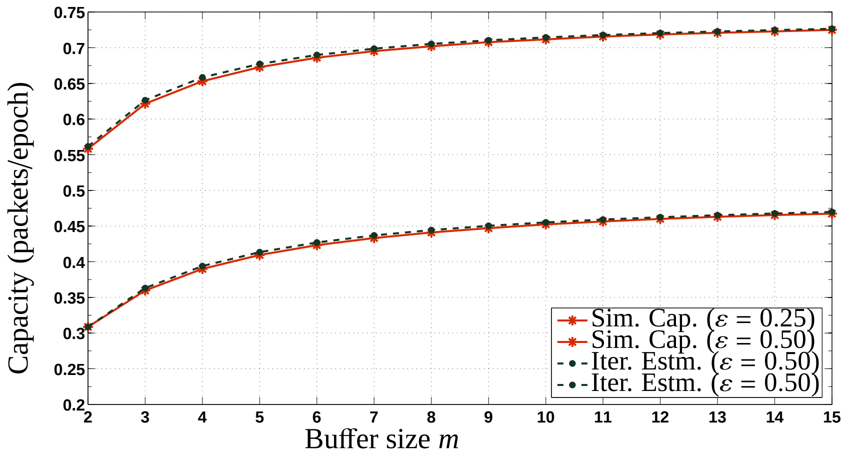

Figure 3.4: Capacity of a line network with $h = 8$ vs. the buffer size m.

Figure 3.5 presents the variation of delay distribution with respect to the buffer size for an eight-hop line network with the erasure probability on every link set to 0.25. It can be seen that both the mean and the variance of the distribution increases with the increase in the buffer size. It is noted that the analytic prediction of the delay is more conservative than the actual simulation i.e., the analytic estimate of the variance is higher than the actual simulated one.

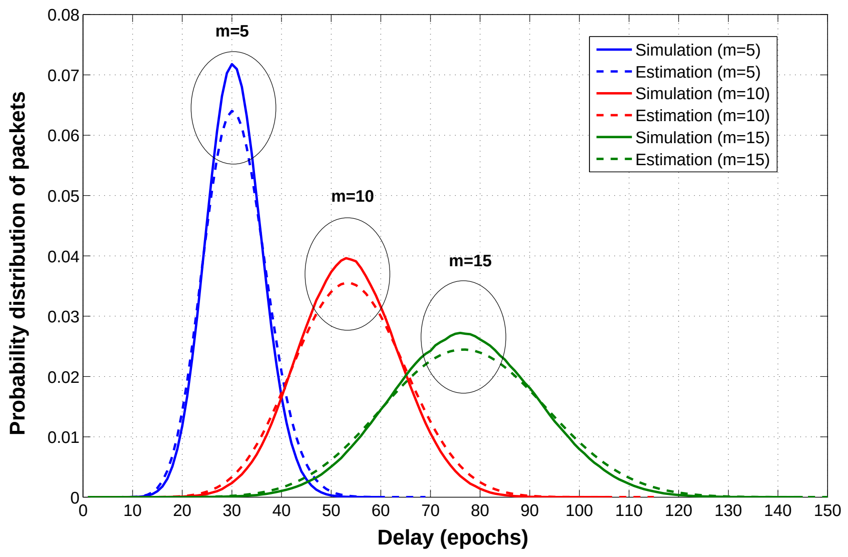

Figure 3.5: Delay distribution in an 8 hop line network for varying buffer sizes.

CHAPTER 4

ANALYSIS OF THROUGHPUT AND DELAY IN WIRED ACYCLIC ERASURE NETWORKS

In this chapter, we apply our proposed iterative method to estimate the performance parameters such as throughput and average latency in general wired acyclic networks with erasure links.[1] As a case study, a random packet routing scheme with ideal feedback on the links is used. We will show that the proposed framework yields a fairly accurate estimate of the probability distribution of buffer occupancies at the intermediate nodes using which we can not only identify the congested and starving nodes but also obtain analytical expressions for throughput and average delay of a packet in the network.

4.1 Network Model and Routing Scheme

We model the network by an acyclic directed graph $\overrightarrow{G}(V, \overrightarrow{E})$, where packets can be transmitted over a link $\overrightarrow{e} = (u, v)$ only from the node u to v. The system is analyzed using a discrete-time model, where each node can transmit at most a single packet over a link in an epoch. The links are assumed to be unidirectional, memoryless and lossy, i.e., packets transmitted on a link $\overrightarrow{e} = (u, v) \in \overrightarrow{E}$ are lost randomly with a probability of $\varepsilon_{\overrightarrow{e}} = \varepsilon_{(u,v)}$. Each node $v \in V$ has a buffer size of m_v packets with each packet having a fixed size. Source and destination pairs are assumed to have sufficient memory to store any data packets. Also, the source node can generate infinitely many packets during each epoch. Node s and node d represent the source and destination nodes respectively. Also, for any $x \in [0, 1]$, $\overline{x} \triangleq 1 - x$.

we consider a directed random routing scheme for packets together with lossless zero-delay feedback on the links. To be more precise, the nodes operate using the following rules, one after another.

[1]This work is done in collaboration with my former lab-mate, Dr. Badri N. Vellambi [72].

1. At each epoch, every node u selects a random ordering of the outgoing edges and transmits the packets it houses one by one. If the packet is successfully received and stored at a neighbor, u deletes the packet from its buffer and transmits the next packet (if any) on the next edge in the selected order. Else, it tries to transmit the same packet on the next (in the selected order) outgoing edge. This process is continued until all packets are transmitted or a transmission is attempted on each link. Therefore, a node with z_o outgoing links transmits at most z_o packets per epoch.

2. After the transmission attempts are made, the node attempts to accept the arriving packets. If more packets are received than it can store, it selects a random subset of the arriving packets whose size equals the amount of space available and stores the selected packets. Consistent with the previous step, appropriate acknowledgment messages are then sent.

4.2 Understanding Finite-Buffer Analysis

Here, we study the tools and steps that enable our framework for analyzing finite-buffer wired acyclic erasure networks. As mentioned in previous chapters, the problem of identifying the throughput and delay is equivalent to the problem of finding the buffer occupancy distribution of the intermediate nodes as a result of ergodicity of the corresponding Markov chain. The routing scheme described in Section 4.1, performs no replication and hence, the buffer state of a node can simply be defined to be the number of physical packets it stores. As seen before, this concept of occupancy follows a Markovian behavior and hence can be studies using our proposed framework.

4.2.1 Approximate Markov Chain for an intermediate Node

Consider a node $u \in V$ in a network $\overrightarrow{G}(V, \overrightarrow{E})$ with z_i incoming and z_o outgoing edges and a buffer size of m_u as depicted in Figure 4.1. Let the nodes that can send packets to u be denoted by $\mathcal{N}^+(u) \triangleq \{v_i, \ldots, v_{z_i}\}$. Similarly, let the nodes to which u can send packets be

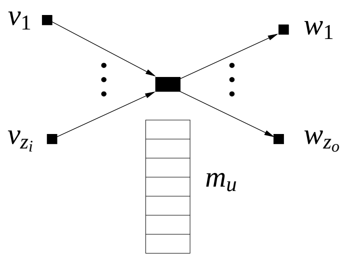

Figure 4.1: A Node in a general wired network.

denoted by $\mathcal{N}^-(u) \triangleq \{w_i, \ldots, w_{z_o}\}$. We assume that the following assumptions hold in the network regarding the arrival and departure processes.

A1. For each $k = 1, \ldots, z_i$, suppose that the packets arrive on (v_k, u) in a memoryless fashion with a rate of λ_k packets/epoch. Also, the processes on different incoming links are statistically independent.

A2. At any instant, for every $k = 1, \ldots, z_o$, a packet is sent on (u, w_k) it is successfully received and stored at w_k with a probability ω_k independent of the past and future events on the edge.

Note that this is hypothetical since in any realistic model of a network, the probability that a packet is successfully transmitted and stored at the next hop depends not only on the channel conditions, but also state of the next-hop node. Since the state of the next-hop node has dependence on its past, the probability of successful receipt can also be expected to have a dependence on its past. In fact this mode of node operation can be replaced by any other scheme that fits into the Markovian set-up of the assumptions above.

At any instant, the number of packets arriving can range from 0 up to z_i and the number of packets departing can range from 0 to z_o. Hence, at each epoch, the state n_u can change to any other in the set $\{n_u - z_o, \ldots, n_u + z_i\} \cap \{0, \ldots, m_u\}$. At any epoch, the probability a_k with which k packets arrive and the probability e_k with which k packets depart are given by

$$A(x) = \sum_{k=0}^{z_i} a_k x^k = \prod_{j=1}^{z_i} (\overline{\lambda}_j + \lambda_j x) \tag{4.1}$$

$$E(x) = \sum_{k=0}^{z_o} e_k x^k = \prod_{j=1}^{z_o} (\overline{\omega}_j + \omega_j x). \tag{4.2}$$

The dynamics of the number of packets stored at u at the l^{th} epoch is a Markov chain that is similar to the one depicted in Figure 4.2.

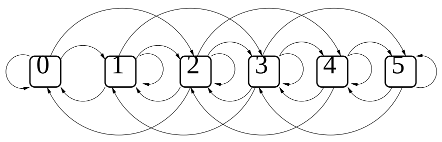

Figure 4.2: The dynamics of a node u with $m_u = 5$ and $z_i = z_o = 2$.

For all input parameters, the Markov chain can be shown to be aperiodic, irreducible and ergodic. Therefore, it possesses a unique steady-state distribution. Letting $\Lambda = (\lambda_1, \ldots, \lambda_{z_i})$ to denote the vector of arrival rates and $\Omega = (\omega_1, \ldots, \omega_{z_o})$ to denote the vector of departure rates, the unique steady-state distribution $\vartheta(\cdot, \Lambda, \Omega, m_u)$ for the chain can be computed using a pair of probability transition matrices T_E and T_A[2] that correspond to the transitions between states that are effected by the departure and arrival of packets, respectively. Note that ϑ is the steady-state distribution after the arriving packets are processed. These transition matrices are defined as follows.

$$T_E = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ \sum_{k=1}^{z_i} e_k & e_0 & 0 & \cdots & 0 & 0 \\ \sum_{k=2}^{z_i} e_k & e_1 & e_0 & \cdots & 0 & 0 \\ \sum_{k=3}^{z_i} e_k & e_2 & e_1 & \cdots & 0 & 0 \\ & & & \vdots & & \\ \sum_{k=m_u}^{z_i} e_k & e_{m_u-1} & e_{m_u-2} & \ldots & e_1 & e_0 \end{bmatrix}. \tag{4.3}$$

[2]For notational consistency, we can extend $e_k = 0$ for $k > z_o$ and $a_k = 0$ for $k > z_i$. Also, for notational convenience, we use $\vartheta(\cdot)$ as a short-hand $\vartheta(\cdot, \Lambda, \Omega, m_u)$.

$$T_A = \begin{bmatrix} a_0 & a_1 & a_2 & a_3 & \cdots & a_{m_u-1} & \sum_{k=m_u}^{z_i} a_k \\ 0 & a_0 & a_1 & a_2 & \cdots & a_{m_u-2} & \sum_{k=m_u-1}^{z_i} a_k \\ 0 & 0 & a_0 & a_1 & \cdots & a_{m_u-3} & \sum_{k=m_u-2}^{z_i} a_k \\ & & & \vdots & & & \\ 0 & 0 & 0 & 0 & \cdots & a_0 & \sum_{k=1}^{z_i} a_k \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix}. \tag{4.4}$$

Note that, the i, jth entry in T_E corresponds to the transition of the occupancy from $i - 1$ to $j - 1$ with the departure of $i - j$ packets. Similarly, the i, jth entry in T_A corresponds to the transition from $i - 1$ to $j - 1$ with the arrival of $i - j$ packets. The actual transition matrix for the Markov chain is then seen to be $T_E T_A$. The steady-state distribution ϑ of the occupancy just after the arriving packets are accepted and the steady-state distribution $\vartheta^\dagger$ of the occupancy just after the packets have been sent but before arriving packets are accommodated are given by

$$\vartheta T_E T_A = \vartheta \text{ and } \vartheta^\dagger T_A T_E = \vartheta^\dagger. \tag{4.5}$$

However, these two steady-state distributions are related by $\vartheta^\dagger = \vartheta T_E$ and $\vartheta = \vartheta^\dagger T_A$. To evaluate the rate of information on the link (u, w_i), one must investigate the rule for packet departure. If at an epoch, more packets are stored than the number of links that allow successful transmission, then each link conveys a packet of information to its neighbors. However, if the occupancy n_u at an epoch l is smaller than the number h of outgoing links that allow for transmission, each link can be assumed to equally receive $\frac{n_u}{h}$ packets on the average – a consequence of the random selection of ordering for outgoing links. Then, the time average of the information rate on the edge (u, w_i) can be seen as

$$\begin{aligned} I(\{(u, w_i)\}, \Lambda, \Omega, m_u) = \sum_{\substack{H \subset \{0,\ldots,z_o\} \\ i \in H}} \Big(\prod_{k \in H} \omega_k\Big) \times \\ \Big(\prod_{k' \in H^c} \overline{\omega}_{k'}\Big)\Big(\sum_{j \geq |H|} \vartheta(j) + \sum_{j < |H|} \frac{j}{|H|} \vartheta(j)\Big). \end{aligned} \tag{4.6}$$

In a similar argument, we notice that some of the arriving packets get randomly blocked if all the arriving packets cannot be stored. We can evaluate the probability with which a packet arriving on the edge (v_i, u) is blocked from

$$\begin{aligned} p_b(\{(v_i,u)\};\Lambda,\Omega,m_u) = \sum_{\substack{H\subset\{0,\ldots,z_i\}\\ i\in H}} \Big(\prod_{k\in H\setminus\{i\}}\lambda_k\Big)\times \\ \Big(\prod_{k'\in H^c}\overline{\lambda}_{k'}\Big)\Big(\sum_{m_u-j<|H|}\frac{|H|-m_u+j}{|H|}\vartheta^{\dagger}(j)\Big). \end{aligned} \tag{4.7}$$

4.2.2 Iterative Estimation of the Buffer Occupancy Distributions

In this section, we discuss our iterative estimation technique in details based on the approximate Markov chain model introduced in Section 4.2.1. Considering that blocking will introduce dependence of the packet incoming/outgoing process over each edge on its past, in order to use the results of Section 4.2.1, we have to make certain simplifying assumptions on the blocking phenomenon. We model the blocking on every edge $\overrightarrow{e} = (u, v)$ of the network as follows.

- Every packet that arrives at v successfully (without getting erased) is blocked in a memoryless fashion with probability q_{uv}. Also, at any epoch, the blocking of packets on any subset of incoming edges of v is assumed to be independent of one another.

Under the above assumption, the blocking process and hence the departure process on every link of the network is modeled as a memoryless process. Since each packet arriving on an edge $\overrightarrow{e} = (u, v)$ is blocked with a probability of q_{uv}, a packet arriving on $\overrightarrow{e}$ is accepted only if both the channel allows the packet and the node accepts it. Therefore, the effective departure rate on the edge (u, v) is seen to be $\overline{\varepsilon}_{uv}\overline{q}_{uv}$. Assuming that the node operates in the mode described in Section 4.1, we can use (4.6) and (4.7) to identify both the rate of information flow and the blocking probabilities on every edge of the network. Thus, the problem reduces to finding a solution $(\varrho_{uv}, q_{uv})_{(u,v)\in\overrightarrow{E}}$ that satisfies the following system of non-linear equations for each $(u, v) \in \overrightarrow{E}$.

$$\varrho_{uv} = \begin{cases} \overline{\varepsilon}_{uv} & u = s \\ \frac{I(\{(u,v)\},(\varrho_{wu})_{w\in\mathcal{N}^+(u)},(\overline{\varepsilon}_{uu'}\overline{q}_{uu'})_{u'\in\mathcal{N}^-(u)},m_u)}{\overline{q}_{uv}} & u \neq s \end{cases},$$

$$q_{uv} = \begin{cases} p_b(\{(u,v)\};(\varrho_{wv})_{w\in\mathcal{N}^+(v)},(\overline{\varepsilon}_{vv'}\overline{q}_{vv'})_{v'\in\mathcal{N}^-(v)},m_v) & v \neq d \\ 0 & v = d \end{cases}.$$

Note that in the above equations ϱ_{uv} represents the fraction of time at which packets will be delivered to v. However, the actual rate of information flow is equal to $\rho_{uv} = \overline{q}_{uv}\varrho_{uv}$.

Finally, the solution to the system of equations can be found by identifying the limit of the sequence defined by the following iterative procedure[3].

1. Set $i = 1$ and for each edge $(u, v) \in \overrightarrow{E}$, set $q_{uv} = 0$ and $\varrho^{(1)}_{uv} = \begin{cases} 0 & u \neq s \\ \overline{\varepsilon}_{uv} & u = s \end{cases}$.

2. Compute $\varrho^{(i+1)}_{uv}, q^{(i+1)}_{uv}$ by using $\varrho^{(i)}_{uv}, q^{(i)}_{uv}$ on the right-hand side of the above system of nonlinear equations and increment i by 1.

3. If $i < L + 1$, perform step 2.

4.3 Estimation of the Throughput and Average Packet Delay

In this section, we exploit the results of the iterative estimation method for buffer occupancy distributions and obtain analytical expressions for throughput and average delay.

Since the routing scheme is such that information is not replicated at any node, the estimate of the total information that arrives at the destination is the sum total of the information rate arriving on each of its incoming edges. Hence,

$$\hat{C}(s, d, \overrightarrow{G}) = \sum_{v\in\mathcal{N}^+(d)} \varrho^*_{vd}(1 - q^*_{vd}) = \sum_{v\in\mathcal{N}^+(d)} \varrho^*_{vd}, \tag{4.8}$$

where we let $(\varrho^*_{uv}, q^*_{uv})$ to be either the component-wise limit of the sequence $\{\varrho^{(i)}_{uv}, q^{(i)}_{uv}\}_{i\in\mathbb{N}}$ when $L = \infty$, or $(\varrho^{(L)}_{uv}, q^{(L)}_{uv})$ when $L < \infty$. Additionally, by the conservation of information

[3]In practice, the number of iterations L which suffice to converge to the solution within reasonable accuracy depends on the structure of the network. Alternatively, one may use $|\vartheta^{(i+1)} - \vartheta^{(i)}| + |\vartheta^{\dagger(i+1)} - \vartheta^{\dagger(i)}| < \epsilon$ for the convergence criteria.

flow, the above estimate can be obtained by computing the rate of flow of information through any cut F using the following.

$$\hat{C}(s, d, \overrightarrow{G}) = \sum_{uv \in F} \varrho^*_{uv}(1 - q^*_{uv}). \tag{4.9}$$

As defined in Section 4.1, the routing scheme assumes feedback on all the links and we treat packets in a First-Come First-Serve (FCFS) fashion at the buffers. Also, the absence of directed cycles allows us to assign an order $v_1, v_2, \ldots, v_n$ to all the nodes of the network in a manner that we have $i < j$ for every link $(v_i, v_j) \in \overrightarrow{E}$.

In order to estimate the average delay that a packet experiences in the network, one can proceed in a recursive fashion. The average delay that an arriving packet (at time l) at node $u \in V$ experiences depends on the buffer occupancy of the node u and its outgoing links. For example, suppose at epoch l (packet arrival time), node u has already k packets where $k \leq m_u - 1$. Then, the arriving packet has to wait for the first k packets to leave node u before it can be transmitted. We define $\mathcal{D}_u(k)$ as the average time it takes from the instant that node u receives a packet given that it has already stored k packets, until the time that the destination node receives that packet. We compute the average delay function $\mathcal{D}_u(.)$ for all the intermediate nodes $u \in V$ using the following proposition.

Proposition 4.1 *Let $r_{uv} = \overline{\varepsilon}_{uv}\overline{q}_{uv}$ be the average packet transfer rate on link $(u, v) \in \overrightarrow{E}$ and r^-_u be the sum of the rates on all outgoing edges (*i.e.*, $r^-_u = \sum_{v \in \mathcal{N}^-(u)} r_{uv}$). Also, let $\pi_v(j)$ ($j = 0, 1, \ldots, m_v - 1$) be the steady state probability of the buffer of node $v \in V$ storing already j packets right before a new packet arrives and is stored in the buffer. For every intermediate node $u \in V$, given the average delay functions of all its next-hop neighbors ($\mathcal{D}_v(j)$ for all $v \in \mathcal{N}^-(u)$ and $j = 0, 1, \ldots, m_v - 1$), $\mathcal{D}_u(.)$ can be obtained by*

$$\mathcal{D}_u(k) = \frac{k+1}{r^-_u} + \sum_{w \in \mathcal{N}^-(u)} \frac{r_{uw}}{r^-_u}\Big(\sum_{j=0}^{m_w - 1} \pi_w(j)\mathcal{D}_w(j)\Big) \tag{4.10}$$

for $k = 0, 1, \ldots, m_u - 1$.

Proof. Equation (4.10) can be interpreted as follows:

1. The first term represents the average time it takes for a total of $k+1$ packets (counting our selected packet) to leave node u successfully.

2. The second term relates to the average delay due to the rest of the network. The probability of conveying a packet from node u to node v can be estimated by $\frac{r_{uv}}{r_{\bar{u}}}$. An arriving packet at node v finds its buffer already occupied by j packets with probability $\pi_v(j)$. Thus, the packet will experience an average delay of $\mathcal{D}_v(j)$ from this node to the destination. Hence, the average packet delay from node v to the destination is equal to $\sum_{j=0}^{m_w-1} \pi_w(j)\mathcal{D}_w(j)$.

It is easy to see that $\pi_v(j)$ can be calculated using

$$\pi_v(j) = \begin{cases} \frac{\vartheta_v^\dagger(j)}{1-\vartheta_v^\dagger(m_v)} & j = 1, 2, \ldots, m_v - 1 \\ 0 & j = m_v \end{cases} \tag{4.11}$$

To obtain the average packet delay from the source to the destination, the average delay function $\mathcal{D}_u(.)$ is computed for all the nodes in the reverse order[4] (*i.e.*,$\{v_n, \ldots, v_2, v_1\}$). Then, the total average packet delay ($\mathcal{D}_s(0)$) is computed by applying Proposition 4.1 to the source node.

4.4 Simulation Results

In this section, we present the results of actual network simulations in comparison with our analysis and will show that our framework gives accurate estimates of buffer occupancy distributions as well as throughput and average delay.

We consider the network shown in Figure 4.3 to compare the results of the simulation and inferences. In this network, all the edges have $\varepsilon = 0.5$ (erasure probability) except the edges $\{(1, 2), (1, 3), (15, 17), (16, 17)\}$ for which $\varepsilon = 0.05$. All the intermediate nodes are assumed to have the same buffer size. In order to measure the exact performance parameters of this network, millions of packets are sent from the source (Node 1) to the destination

[4]Note that we have $\mathcal{D}_d(k) = 0$ for every k where d denotes the destination node.

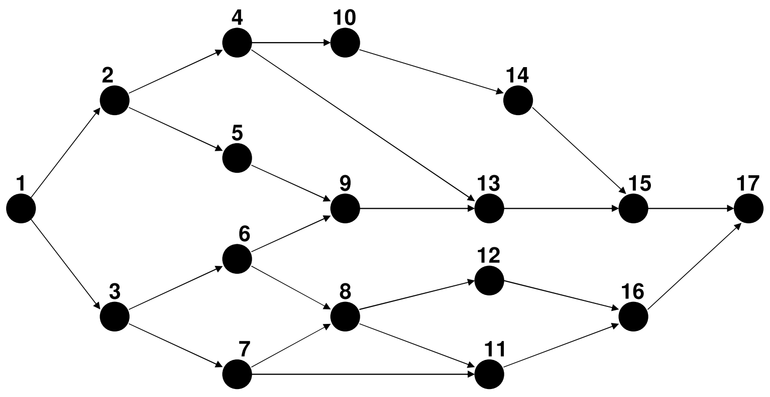

Figure 4.3: A general wired acyclic directed network chosen for simulation).

(Node 17). Figure 4.4 presents a comparison between the actual buffer occupancy dis-

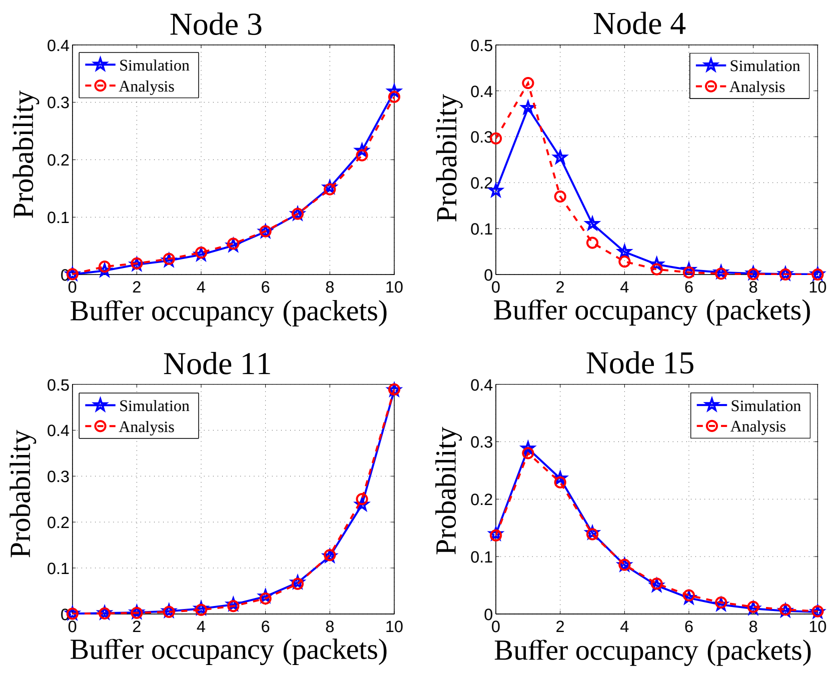

Figure 4.4: buffer occupancy distributions for nodes 3, 4, 11 and 15.

tributions and our iterative estimates for four of the nodes in the network of Figure 4.3. Also, Figure 4.5 presents the variations of the actual throughput and average packet delay and our analytical results versus the buffer size. Note that, the throughput is presented in *packets/epoch* and average packet delay is presented in *epochs*. As it can be observed, our estimation is very close to the actual simulation results.

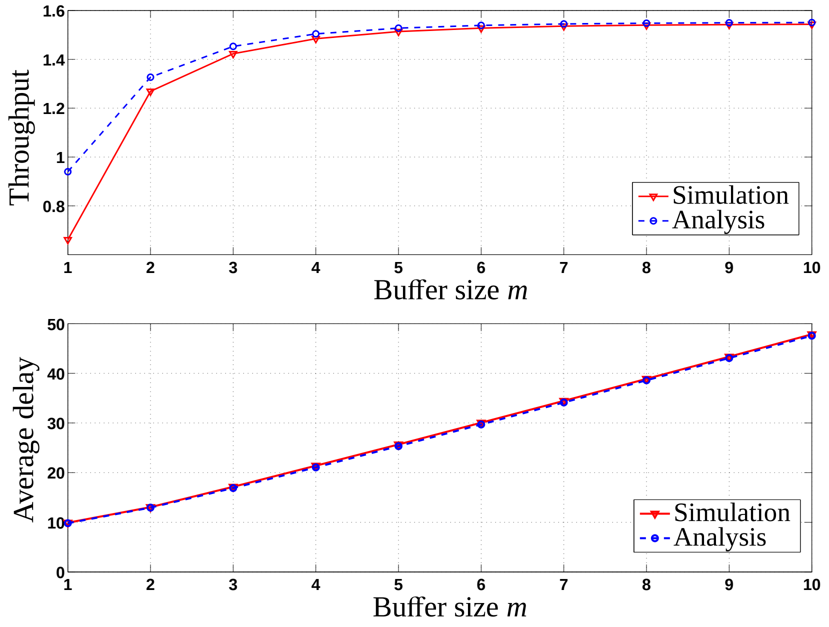

Figure 4.5: Performance parameters of the network for different buffer sizes.

CHAPTER 5

PERFORMANCE OF WIRELESS ERASURE NETWORKS WITH BACKPRESSURE ROUTING

In this chapter, we focus on the problem of performance analysis in wireless erasure networks and investigate the trade-offs between throughput, average packet delay and buffer size when a modified backpressure routing policy is used. Our approach employs a discrete-time model to approximate the buffer occupancy distributions at the intermediate nodes. We then obtain analytical expressions for throughput and average packet delay in terms of the estimated buffer occupancy distributions.

5.1 Network Model and Routing Scheme

We adapt the wireless model used in [49, 63]: For any node $v \in V$ with multiple outgoing links, by the broadcast property of the wireless medium, the same packet is sent over all the outgoing links at the same time epoch t (t is an integer). Further, multiple arriving packets for a node $u \in V$ from different incoming links do not interfere and can be stored in a single epoch[1] if there is enough space available in the buffer of node u. In case there is not enough space available in the buffer, some of the arriving packets will be randomly *blocked* by node u. Further, at each epoch, we assume the transmission of a single packet by every node.

Here, our goal is to analyze the performance of Diversity Backpressure Routing (DIVBAR) [63]. DIVBAR is generally desirable because of its flexible approach which can dynamically adjust routing decisions in response to the random outcome of the transmissions. In this scheme, every node $u \in V$ transmits a packet in each epoch (blind packet transmissions). After receiving ACK/NACK feedbacks from the various receivers $\mathcal{R} \subset \mathcal{N}^{-}(u)$, node u chooses the receiver node $v \in \mathcal{R}$ with the largest positive differential backlog (*i.e.*,

[1]Interferences are avoided in such environments using some form of time, frequency or code division multiple access schemes.

$Q_u(t) - Q_v(t)$) to take the responsibility of forwarding the packet on the path. Here, the backlog parameter $Q_u(t)$ is defined as the current number of packets stored in any node u at the beginning of the time epoch t. Next, node u and all the other receivers delete the packet from their buffers. The algorithm, also breaks ties arbitrarily and retains the packet in u if no receiver has a positive differential backlog. Note that the backlog parameter of each receiver can simply be included in the ACK/NACK signal to be sent back to node u. Note that, the routing scheme is asymptotically throughput optimal meaning that it achieves the wireless min-cut capacity [49] when the buffer sizes are sufficiently large. However, here we only aim to study the interplay of throughput and average latency achieved by the back-pressure routing in finite-buffer regime.

5.2 Markov Chain Modeling

Thoroughly investigated in [29] for the exact analysis of a finite-buffer line network, as a result of ergodicity of the corresponding MCs, the problem of identifying the throughput is equivalent to the problem of finding the buffer occupancy distribution of the intermediate nodes. Further, due to the exponential growth in the size of the exact Markov chain, exact calculation of the steady-state probability distributions of the buffer occupancies and the network performance is computationally intractable even for networks of reasonable size[2]. Hence, we propose an approximation method that for every node $u \in V$ updates its queue ($Q_u(t)$) considering: 1. The probability of packet arrival at u from the previous-hop neighbors $\mathcal{N}^{+}(u)$, and 2. The effect of blocking imposed by the next-hop neighbors $\mathcal{N}^{-}(u)$. Hence, we will only consider the dependency of the state transition probabilities of the queue for each node u to the state of the queues corresponding to nodes in $\mathcal{N}^{+}(u)$ and $\mathcal{N}^{-}(u)$. Moreover, the main idea of the approximation framework is to divide the multi-dimensional MC with multiple reflections into multiple simple MCs (*i.e.*, Only $Q_u(t)$ for every node $u \in V$) whose steady-state probability distributions can be calculated separately

[2]For a network of N intermediate nodes, the exact MC has $(m+1)^N$ states

in terms of the steady state probability distributions of the other related MCs. Note that although each MC process is assumed independent of the other MC processes, the interdependency of the states of their queues are captured by the approximation method via their state transition probabilities.

Consider a node $u \in V$ in a network $\overrightarrow{G}(V, \overrightarrow{E})$ with d_i incoming and d_o outgoing edges and a buffer size of m. Let $\mathcal{N}^+(u) \triangleq \{v_1, \ldots, v_{d_i}\}$ and $\mathcal{N}^-(u) \triangleq \{w_1, \ldots, w_{d_o}\}$. By means of our approximation method, the state transition probabilities for MC of any queue $Q_u(t)$ depend on the steady-state distributions of the queues of nodes in $\mathcal{N}^+(u)$ and $\mathcal{N}^-(u)$. Since there is no prior information about the probability distribution of these queues, the proposed estimation algorithm must be performed iteratively. To determine the state transition probabilities for each MC, we need to know the dynamics of arrival and departure of packets to its corresponding queue. As a result of our approximation assumptions, for every queue $Q_u(t)$, we define multiple incomming and outgoing streams of packets which are assumed to be statistically independent. In our model, since we allow the reception of multiple packets in an epoch, the number of arriving streams is the same as the number of incoming links to a node. Also note that, the occupancy of node u ($Q_u(t)$) directly affects the arrival rates, since the probability that node u is selected as the receiver with the largest positive differential backlog is higher when $Q_u(t)$ is smaller. Further, as a result of the broadcast property, only one packet can be conveyed to the set of receivers which implies that there is only one departing stream. In a similar argument, $Q_u(t)$ has a considerable effect on the departure rate since the expected number of receivers with positive differential backlog increases with $Q_u(t)$.

As a result, given $Q_u(t) = n_u$ for an arbitrary node u, we define the set of arrival rates as $\Lambda_u = \{\lambda_1(n_u), \ldots, \lambda_{d_{in}}(n_u)\}$ and the departure rate as $\Omega_u = \{\mu(n_u)\}$. In other words, $\lambda_i(n_u)$ is the probability of an arrival of a packet at node u coming from node v_i (*i.e.*, shifting the responsibility of forwarding a packet from v_i to u) given there are already n_u packets stored at u. Similarly, $\mu(n_u)$ is the probability of a departure of a packet from node u to one of its receivers in $\mathcal{N}^{-}(u)$. Further, for the systematic representation, we define the arrival and departure polynomials by

$$\begin{aligned} A^{(n_u)}(x) &= \sum_{k=0}^{d_{in}} a_k^{(n_u)} x^k = \prod_{j=1}^{d_{in}} (\overline{\lambda_j(n_u)} + \lambda_j(n_u)x), \\ E^{(n_u)}(x) &= e_0^{(n_u)} + \overline{e_0^{(n_u)}} x = \overline{\mu(n_u)} + \mu(n_u)x. \end{aligned}$$

Let $\Delta_u = \{A^{(n_u)}(x)\}_{n_u=0}^{m}$ and $\Gamma_u = \{E^{(n_u)}(x)\}_{n_u=0}^{m}$ be the sets of all arrival and departure polynomials for the queue $Q_u(t)$, respectively. Given Δ_u and Γ_u, the state transition probabilities of the MC for $Q_u(t)$ can be easily computed. As an example, for $0 < j < m$, we have the following[3]:

$$Pr\{Q_u(t+1) = j | Q_u(t) = i\} = \sum_{k=0}^{d_{in}} a_k^{(i)} e_{k+i-j}^{(i)} .$$

As a result, the proper approximate MC is formed for $Q_u(t)$ with steady-state probability distribution denoted by $\pi_u(\cdot)$.

In summary, for every node u in network, a queue with its incoming and outgoing streams will be identified properly. Then, the corresponding arrival and departure polynomials will be obtained parametrically. These polynomials describe the state transitions of the queue from which the steady-state probability distribution can be computed for the MC. Then, we apply IEA to compute steady-state probability distributions for all the nodes as follows:

Step 1. Initialization (iteration 0): Start with arbitrary rates for the arrival/departure in every node u (*i.e.*, $\Lambda_u^{(0)}$ and $\Omega_u^{(0)}$) and compute $\Delta_u^{(0)}$ and $\Gamma_u^{(0)}$ for each queue, where the superscript denotes the iteration number. However, apply prior information regarding the queues for initialization. For example, in our model, destination node does not block any arriving packet. Also, the source node has infinitely many packets.

Step 2. Increase the iteration index by one (e.g., iteration i). Given $\Delta_u^{(i-1)}$ and $\Gamma_u^{(i-1)}$ for every node u, compute their steady-state probability distributions $\pi_u^{(i-1)}(\cdot)$.

[3]For notational consistency, we can extend $e_k = 0$ for $k < 0$ or $k > 1$ and $a_k = 0$ for $k < 0$ or $k > d_{in}$.

Step 3. Given $\pi_u^{(i-1)}(\cdot)$ for all the MCs, compute the new sets of arrival/departure polynomials $\Delta_u^{(i)}$ and $\Gamma_u^{(i)}$ for every node u.

Step 4. Go back to Step 2 until all the steady-state probabilities converge to fixed distributions[4].

Note that at Step 3, given the steady state probability distribution for every node, we need to find the arrival/departure polynomials for all the queues. First, we notice that some of the arriving packets on the link (v_i, u) get blocked randomly when all the arriving packets cannot be stored due to node u's state full buffer at time t. Given Δ_u and $Q_u(t) = n_u$, the blocking probability on the link (v_i, u) can be evaluated using

$$p_b\{(v_i,u)|n_u\}=\sum_{H\subset\{0,\dots,d_i\}\backslash\{i\}}\Big(\prod_{k\in H\backslash\{i\}}\lambda_k(n_u)\Big)\Big(\prod_{k'\in H^c}\overline{\lambda_{k'}(n_u)}\Big)\max\left\{\frac{n_u+|H|+1-m}{|H|+1},0\right\}.$$

Similarly, a packet is "conveyed" over the link (u, w_i) only when it is not erased on the link and node w_i has the largest positive differential backlog with respect to node u in comparison to all the other successful recipients of the packet at an epoch. Then, given Γ_u and $Q_{w_i}(t) = n_{w_i}$, the rate with which the packets are conveyed over the link (u, w_i) can be obtained as

$$I\{(u,w_i)|n_{w_i}\}=\sum_{l=n_{w_i}+1}^{m}\pi_u(l)\sum_{H\subset\{0,\dots,d_o\}}\left(\prod_{k\in H}\overline{\varepsilon_{(u,w_k)}}\right)\left(\prod_{k'\in H^c}\varepsilon_{(u,w_{k'})}\right)\left(\sum_{Q\subset H}\frac{\prod_{q\in Q\backslash\{i\}}\pi_{w_q}(n_{w_i})\prod_{q'\in Q^c}\Big(\sum_{j=n_{w_i}+1}^{m}\pi_{w'_q}(j)\Big)}{|Q|}\right).$$

Finally, given $Q_u(t) = n_u$ the arrival/departure rates can be obtained by

$$\begin{aligned}\mu(n_u) &= 1-\sum_{Q\subset\{1,\dots,d_{out}\}}\Big(\prod_{k\in Q}\overline{\varepsilon_{(u,w_k)}}\Big)\Big(\prod_{k'\in Q^c}\varepsilon_{(u,w_k)}\Big)\Big(\prod_{q\in Q}\Big(\sum_{i=0}^{n_u-1}\pi_{w_q}(i)p_b\{(u,w_q)|i\}+\sum_{j=n_u}^{m}\pi_{w_q}(j)\Big)\Big),\\ \lambda_k(n_u) &= I\{(v_k,u)|n_u\}.\end{aligned}$$

[4]Convergence of the steady-state probabilities is measured by checking the distance between their estimates for two consecutive iterations and stopping the iterations when the distance becomes less than a certain threshold.

5.3 Estimation of the Throughput and Average Packet Delay

In this section, we exploit the resulting buffer occupancy distributions and obtain analytical expressions for throughput and average delay. Since the information rate (The rate of conveying packets) on different links are independent, the throughput estimate $\hat{C}(s, d, \overrightarrow{G})$ from the source node s to the destination node d is the sum of the information rates arriving to the destination node. Hence,

$$\hat{C}(s, d, \overrightarrow{G}) = \sum_{v \in \mathcal{N}^+(d)} I\{(v, d)\}.$$

In order to estimate the average delay, one can proceed in a recursive fashion. The average delay that an arriving packet at node $u \in V$ experiences depends on the buffer occupancy of the node u as well as the dynamics of its packet departures. For example, suppose at epoch t (packet arrival time), node u has already n_u packets where $n_u \leq m - 1$. Then, the arriving packet has to wait for the first n_u packets to leave node u before it can be served. We define $\mathcal{D}_u(n_u)$ as the average time it takes from the instant that node u stores an arriving packet at time t when $Q_u(t) = n_u$ until the time that the destination node receives that packet. Further, Let $\mathcal{L}_u(x, y)$ be the average delay for y packets to depart from node u given it has already x packets in its buffer, $x \geq y$. In order to obtain $\mathcal{D}_u(n_u)$, first we need to compute $\mathcal{L}_u(x, y)$ by solving the corresponding transient MC using the following lemma.

Lemma 5.1 *Let $T^{(0)}$ and $T^{(1)}$ be $(m + 1) \times (m + 1)$ matrices, where $T^{(1)}_{i,j}$ is the transition probability from state $\{i - 1\}$ to state $\{j - 1\}$ for $Q_u(t)$ when a single departure occurs, and $T^{(0)}_{i,j}$ is the transition probability from state $\{i - 1\}$ to state $\{j - 1\}$ when no departure occurs. Then, given $\mathcal{L}_u(x, 0) = 0$, for $x = 1, \ldots, m$ and $y \leq x$, we have*

$$\mathcal{L}_u(x, y) = \frac{1}{1 - T^{(0)}_{x+1,x+1}} \left(1 + \sum_{i=1}^{m-x} T^{(0)}_{x+1,x+1+i} \mathcal{L}_u(x + i, y) + \sum_{j=-1}^{m-x} T^{(1)}_{x+1,x+1+j} \mathcal{L}_u(x + j, y - 1) \right).$$

Next, using Lemma 5.1, we compute $\mathcal{D}_u(\cdot)$ for all the intermediate nodes $u \in V$ by the following proposition.

Proposition 5.1 *Let $\phi_v(n_v)$ for $n_v = 0, 1, \ldots, m-1$, be the steady state probability of node $v \in V$ storing n_v packets right before it stores a new arriving packet. In other words, $\phi_v(n_v)$ is the conditional probability of the event $Q_v(t) = n_v$ given that $Q_v(t) < m$. Also, let $I\{(u,v)\} = \sum_{n_v=0}^{m-1} I\{(u,v)|n_v\}\pi_v(n_v)$. Given $\mathcal{D}_v(\cdot)$ for all nodes $v \in \mathcal{N}^-(u)$, for every node $u \in V$, we can obtain $\mathcal{D}_u(n_u)$ for $n_u = 0, 1, \ldots, m-1$ using:*

$$\mathcal{D}_u(n_u) = \mathcal{L}_u(n_u+1, n_u+1) + \sum_{w \in \mathcal{N}^-(u)} \sum_{n_w=0}^{m-1} \frac{I\{(u,w)|n_w\}}{\sum_{v \in \mathcal{N}^-(u)} I\{(u,v)\}} \phi_w(n_w)\mathcal{D}_w(n_w) \tag{5.1}$$

Proof. Equation (5.1) can be interpreted as follows:

1. The first term represents the average time it takes for a total of $k+1$ packets (counting the selected subject packet) to leave node u successfully (and to be stored at one of the next-hop nodes) which is obtained using Lemma 5.1.

2. The second term relates to the average delay due to the travel of the packet through the rest of the network. The probability of conveying a packet from node u to node w can be estimated by $\frac{I\{(u,w)|n_w\}}{\sum_{v \in \mathcal{N}^-(u)} I\{(u,v)\}}$. An arriving packet at node w finds its buffer already occupied by n_w packets with probability $\phi_w(n_w)$. Thus, the packet will experience an average delay of $\mathcal{D}_w(n_w)$, computed from this node to the destination. Hence, the average packet delay computed from node v to the destination is equal to $\sum_{n_w=0}^{m-1} \frac{I\{(u,w)|n_w\}}{\sum_{v \in \mathcal{N}^-(u)} I\{(u,v)\}} \phi_w(n_w)\mathcal{D}_w(n_w)$.

Finally, the total average packet delay, $\mathcal{D}_s(0)$, can be computed by applying Proposition 5.1 to the source node.

5.4 Simulation Results

In this section, we present the results of actual network simulations in comparison with our analysis and show that our framework gives accurate estimates of throughput and average delay.

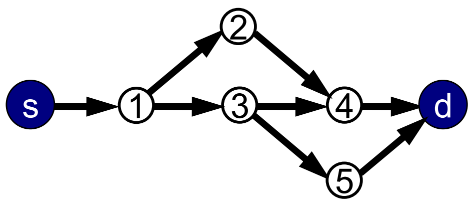

Figure 5.1: A sample network.

Consider the sample network in Figure 5.1. In this network, all the erasure probabilities are chosen to be 0.5 except for the link $(s, 1)$ which is chosen to be 0.05. Figure 5.2 presents the variation of our analytical results and the actual simulations for both throughput and average latency, as the buffer size m is varied. Note that, the throughput is presented in *packets/epoch* and average packet delay is presented in *epochs*. It is noticed that the iterative estimate accurately captures the variation of the performance parameters obtained by simulations. It can be seen that as the buffer size is increased, all curves approach the wireless min-cut capacity [49] of $1 - \varepsilon_{(1,2)}\varepsilon_{(1,3)} = 0.75$. An important observation in this example is that increasing the buffer size beyond $m = 5$ does not improve the throughput significantly. However, it dramatically increases the average latency. This implies that even if a large buffer is available at nodes, it is not a good idea to allocate more than about 5 packets to the same flow. Finally, It can be observed that for $m = 1$ the estimations are not as accurate as the ones for other buffer sizes. The reason could be the separation of dependent MCs as a part of our approximation assumptions mentioned in Section 5.2.

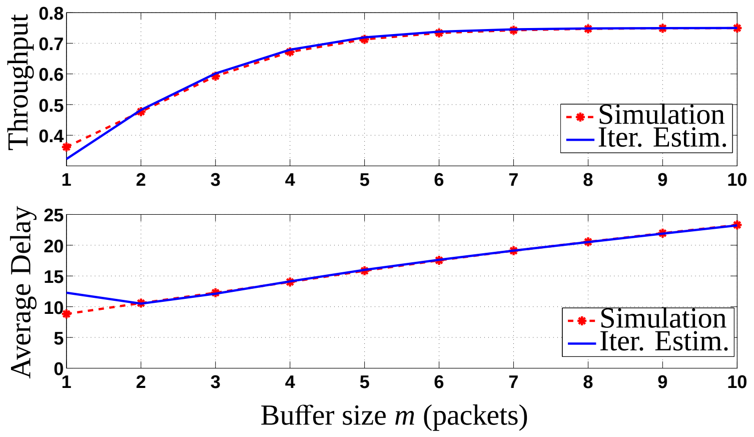

Figure 5.2: Throughput and Average packet delay for the sample network.

CHAPTER 6

BUFFER SIZE OPTIMIZATION FOR DELAY-SENSITIVE APPLICATIONS IN WIRELESS NETWORKS

In this chapter, we study the effect of finite buffer size on the performance parameters of multihomed wireless networks and address the problem of buffer size optimization to meet the requirements of delay-sensitive applications. We use the generalized approximation framework developed in Chapter 2 for analysis which provides an iterative estimation for the distribution of buffer occupancies. We then obtain analytic expressions for the throughput and delay distribution of packets in the network. Finally, using the analytic results, we propose an optimization algorithm to maximize the throughput while bounding the packet delay to an application-dependent threshold for an arbitrarily large portion of the packets.

6.1 Introduction and Motivation

Wireless local area networking (WLAN) is a commonly used technology today. Although there are many options for wide area network (WAN) connections such as ad-hoc networking, many organizations and schools today, provide WLAN access points (AP) for their employees and students to be able to connect to a wired backbone network such as the Internet. While the current Internet traffic mostly consists of Web and email, real time applications such as IPTV and VoIP are becoming increasingly important. Such applications are fundamentally different from data traffic in their sensitivity to delay and loss which has led to a great interest in addressing quality of service (QoS) for delay-sensitive traffic.

There are often multiple APs connected to the Internet backbone through a gateway [73]. The gateway can be a WLAN controller for WiFi networks or a Serving GPRS Support Node (SGSN) in cellular network. The gateway divides the traffic among different APs to reduce the effects of the wireless channel erasures. This is called multihoming which is intended to increase the reliability of network but it does not necessarily improve their performance. Given the wireless channel erasures, the packets may have to be stored at the APs for transmission at a later time. If an unlimited buffer is available, the APs do not have to reject or drop any arriving packets. However, there are several disadvantages for large buffers [2]: 1. Over-buffering increases end-to-end delay, conflicting with real-time applications such as online gaming, IPTV, and VoIP. Additionally, large buffers can increase the variance of the latency, making congestion control algorithms unstable. 2. Large buffers lead to high power consumption, and more board space. Therefore, although increasing the buffer sizes tends to increase the link utilization (and hence throughput), it also tends to increase the queuing delay. Hence, a fundamental question is: what is the buffer requirement given certain constraints on the throughput and delay?

Despite extensive studies of buffer sizing for wired networks and its optimization [74–76], there have been very limited works on the impact of buffer sizes on wireless network performance [77–79]. In [77], a loose upper bound for achievable capacity is used to size the buffers in a wireless mesh network. Further, a dynamic buffer sizing protocol is presented in [79] to lower the delay in 802.11-based WLANs which is not necessarily optimal. Compared to sizing buffers in wired routers, a number of fundamental new issues arise when considering wireless networks. For example, in addition to link utilization (which is a determining factor in buffer sizing of wired networks), the packet loss rate due to wireless channels is also an important metric to consider. Wireless networks are also throughput constrained, and hence buffer sizing can have a profound impact on the application performance. Moreover, new-age applications such as online gaming, IPTV, and VoIP place strict requirements on latency, and hence maximizing the throughput alone is not a sufficient objective. We depart from the past work not only because of our focus on wireless networks but also because of achieving different objectives. The main goals of previous works are to size buffers either to maximize the throughput or to minimize the blocking probability. To the best of our knowledge, no work has considered an analytical sizing of the buffers for maximizing throughput while meeting the delay demands of the application.

There are two important factors to take into account: 1. A fixed buffer size is problematic in wireless networks and may not maximize the link utilization (and hence the throughput), and 2. Some applications would require the enforcing of strict delay requirements. Our work is motivated by such concerns. In other words, we would like to choose buffer sizes adaptively while maximizing the goodput (defined as the throughput that meets the latency requirements on the packets). Several fundamental questions consequently arise: What is the throughput and capacity of finite-buffer wireless networks? What is the delay distribution in such networks? What is the interplay of the latency, throughput and buffer size? Some of these questions have been addressed in the previous chapters. However, computing the delay distribution of packets in a general finite-buffer network has not been addressed. Thus, our objective is to develop a general framework for adaptive buffer sizing. This requires the study of delay distribution and throughput of wireless networks as a function of buffer sizes. We believe that the developed framework can help to understand, design, and analyze practical delay constrained networks, and to design more suitable protocols for real-time applications. Our contributions in this chapter can be summarized as follows:

- Derivation of analytical expressions for the important performance parameters such as network throughput and delay distribution of packets in terms of the buffer occupancy distributions for the case of multihomed wireless LAN topology.

- Applying the analytical estimation results to develop optimization procedures for buffer sizing in wireless networks.

6.2 Problem Statement and Network Model

We can model a multihomed finite-buffer wireless LAN by a directed graph with a source s which represents the gateway, n intermediate nodes $v_1, \ldots, v_n$ with buffer sizes $m_1, \ldots, m_n$ representing the access points and a destination node d representing the user. Further, the destination node has no buffer constraints. The set of lossy links are $\{(s, v_i) : i = 1, \ldots, n\}$ and $\{(v_i, d) : i = 1, \ldots, n\}$. Also, packets are admitted from the wired backbone network into the gateway in a memoryless fashion with mean arrival rate R_{in}. The packet admission rate R_{in} will be used later to control the trade-off between throughput and delay. Finally, at each epoch, s transmits at most one packet to each of the APs and each AP transmits at most one packet to the destination. In our model, multiple arriving packets to a node from different incoming links do not interfere and can be stored in a single epoch. The throughput of a scheme is defined to be the rate of transmission of information packets (in packets per epoch) between the source and destination nodes for a fixed queue management and communication scheme. Further, we define delay or latency of a packet as the time taken from the instant when the source stores the packet to the instant when the destination node receives it. Finally, we define goodput the same as throughput but considering only those packets that met the latency requirements.

We employ the following notations: $\mathbb{G}(p)$ denotes the geometric distribution with mean inter-arrival time $\frac{1}{1-p}$. Further, Let $\mathcal{N}^{-}(u)$ denote the set of all the neighboring nodes that u can send packets via outgoing links. Likewise, the set $\mathcal{N}^{+}(u)$ is defined using the incoming links of u. For any $x \in [0, 1]$, $\overline{x} \triangleq 1 - x$. The convolution operator is denoted by $\otimes$.

For a network with a set of buffer sizes $\mathcal{M}$, set of erasure probabilities $\mathcal{E}$, and packet admission rate R_{in}, the throughput between the source s and the destination d, is denoted by $\mathcal{T}_{s,d}(\mathcal{E}, \mathcal{M}, R_{in})$. Further, the steady-state probability distribution of packet delay, is denoted by $\Pr\{\mathcal{D} = k\}$ for $k = 1, 2, \ldots$, where $\mathcal{D}$ is the random variable representing the delay of a packet. Delay constraints may take various forms depending on the application. Here, as the delay constraint, we assume that at least a certain fraction δ of packets (where $0 < \delta < 1$) is required to reach the destination with a delay smaller than an application-dependent threshold value Δ. Finally, the buffer sizing problem is formulated as

$$\begin{aligned} \underset{\mathcal{M}, R_{in}}{\text{maximize}} \quad & \mathcal{T}_{s,d}(\mathcal{E}, \mathcal{M}, R_{in}) \\ \text{subject to} \quad & \Pr\{\mathcal{D} \leq \Delta\} = f(\mathcal{M}, R_{in}) \geq \delta, \end{aligned}$$

where $\Pr\{\mathcal{D} \leq \Delta\}$ is a function of node buffer sizes and the arrival rate at the source (i.e., the wireless gateway).

6.3 Finding the Performance parameters

In this section, we use the results of Chapter 2 and assume that the approximate buffer occupancy distributions are obtained from the iterative estimation algorithm. Next, using the buffer occupancy estimates, we derive analytic expressions for network throughput and packet delay distribution in a multihomed finite-buffer wireless LAN scenario.

In order to apply the framework of Chapter 2 to this setting, we only need to identify the set of queues and their incoming and outgoing streams. As mentioned before, since the routing scheme is single-copy and independent of buffer occupancies, the buffer states are simply defined as the number of packets each node possesses and are denoted by X_s and X_{v_i} for $i = 1, \ldots, n$. The arrival/departure streams for each queue is also clearly via their incoming/outgoing links besides the gateway which has an exogenous bernoulli packet arrival stream. Therefore, using the proposed framework, we assume the steady-state probability distribution of buffer occupancies can be easily obtained and are denoted by $\pi_s(\cdot)$ and $pi_{v_i}(\cdot)$ for $i = 1, \ldots, n$.

Given the occupancy distributions, the network throughput can be easily derived from

$$\mathcal{T}_{s,d} = \sum_{i=1}^{n} \overline{\varepsilon_{v_i,d}}(1 - \pi_{v_i}(0)). \tag{6.1}$$

This is because the probability of receiving a packet from any of the APs is equal to the probability of the event that it has at least one packet in its buffer and the channel does not erase that packet.

In order to compute the packet delay distribution, we proceed in a hop-by-hop fashion. The additional delay of an arriving packet at node v_i depends on the occupancy of the node and the erasure channel that follows it to the destination. Suppose node v_{h-1} has $k \leq m_i - 1$ packets in addition to the arriving packet. Then, the packet has to wait for the first k packets to leave before it can be serviced. Since each transmission takes place independently, the distribution of delay is sum of $k+1$ independent geometric distribution with mean inter-arrival time $\frac{1}{1-\varepsilon_{v_i,d}}$, which is denoted by $\otimes^{k+1}\mathbb{G}(\varepsilon_{v_i,d})$. Given the occupancy distributions $\pi_{v_i}(\cdot)$ for $i = 1, \ldots, n$, the distribution of delay added by the access points is

$$\mathbf{D}_{AP} = \sum_{i=1}^{n} P_i \sum_{k=0}^{m_i-1} \pi_{v_i}(k) \otimes^{k+1} \mathbb{G}(\varepsilon_{v_i,d}), \tag{6.2}$$

where, P_i is the probability of the event that a packet is routed through node v_i and can be obtained from

$$P_i = \frac{\overline{\varepsilon_{s,v_i}}\,\overline{\pi_{v_i}(m_i)}}{\sum_{j=1}^{n} \overline{\varepsilon_{s,v_j}}\,\overline{\pi_{v_j}(m_j)}}. \tag{6.3}$$

Further, in a similar fashion, the distribution of delay added by the source is

$$\mathbf{D}_S = \sum_{k=0}^{m_s-1} \pi_s(k) \{\otimes_{i=1}^{n} \mathbb{G}(\varepsilon'_{s,v_i})\}^{k+1}, \tag{6.4}$$

where, ε'_{s,v_i} represents the effective erasure of the link (s, v_i) by taking care of the blocking probability of node v_i.

By assuming that the delays incurred by each node and its adjoining outgoing links are independent of each other, we obtain the total delay distribution to be

$$\mathbf{D} = \mathbf{D}_S \otimes \mathbf{D}_{AP}.$$

6.4 Buffer Sizing for Wireless Networks

Thus far, we established a framework to estimate the network performance parameters such as throughput and delay distribution in a finite-buffer regime for an arbitrary given set of buffer sizes. In this section, we use the obtained estimates for delay distribution and throughput (as functions of the buffer sizes) in an optimization problem to answer the buffer sizing questions.

6.4.1 Optimization Algorithm

As formulated in Section 6.2, Our goal is to find the optimal buffer sizing for the entities in wireless networks such that the goodput is maximized (for a fixed delay constraint).

We approach this optimization problem by dividing it into two stages. First, for any given set of buffer sizes $\mathcal{M}$, the maximum admission rate R_{in} can be easily found to meet the delay constraint. This is because: (1) $f(\mathcal{M}, R_{in})$ is a non-decreasing function of R_{in}, and (2) the maximum throughput that satisfies the delay constraint is resulting from the maximum arrival rate R_{in} that meets the delay constraint. Next, a greedy search algorithm is used to find the optimal choices of buffer sizes $\mathcal{M}$. This algorithm leads to the maximum throughput while meeting the delay constraint if a global maximum exists for the throughput. We claim the existence of such a condition since: (1) the throughput is a non-decreasing concave function of the buffer sizes [80], and (2) the amount of back-off from the maximum throughput to meet the delay constraint is a non-decreasing function of buffer sizes $\mathcal{M}$. Note that the amount of back-off from the maximum throughput is closely related to the optimal choice of R_{in} which approaches a constant value for sufficiently large buffer sizes.

6.4.2 Simulation Results

We demonstrate the optimization process by a toy example. Consider a network where packets are admitted through node s (wireless gateway) that is connected to an access point with buffer size m through a wired link with a packet erasure rate of 0.05. Further a mobile node d receives the packets through a wireless link with an erasure rate of 0.9 from the access point. Assume the objective is to maximize the throughput while more than 95% of the packets reach to the destination with a delay smaller than 150 epochs. Using our analytical framework, we start from a random buffer size $m = m^*$ and find the corresponding maximum admission (arrival) rate R_{in} that meets the delay constraint. This admission rate is then used to analytically compute the throughput for $m = m^*$, which is the maximum possible throughput to meet the delay constraint given the buffer size $m = m^*$. Next, we employ a greedy search algorithm by increasing or decreasing m and similarly finding the corresponding maximum delay-constrained throughput until an optimal value for m is determined. The variations of the goodput with buffer size is depicted in Figure 6.1

for both simulation and the analytical estimation of the parameters. It is also observed that the optimal goodput of 0.088 packets per epoch is achieved at $m = 9$ which potentially improves the network goodput by 15%. Note that the maximum achievable throughput (with no delay constraint) is equal to 0.1 which would be obtained by using infinite buffer sizes.

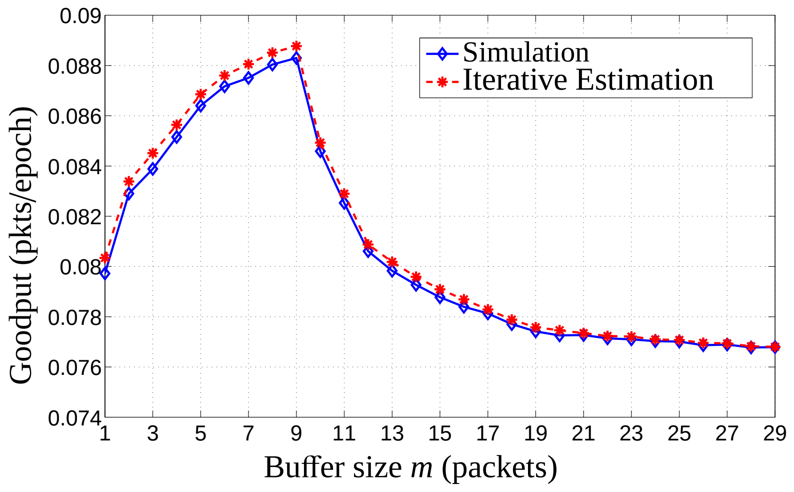

Figure 6.1: Maximized delay-constrained network goodput for varying buffer sizes.

Next, another example is considered in which two APs are connected to the gateway through wired links with a packet erasure rate of 0.05. A mobile node d can also receive packets through wireless links with erasure rates of 0.6 and 0.8 from AP 1 and AP 2, respectively. In this case, the objective is to maximize the throughput while more than 90% of the packets reach to the destination with a delay smaller than 50 epochs. We repeat the same optimization procedure described above and obtain the optimal goodput of 0.584 packets per epoch at non-trivial buffer sizes $(m_1, m_2) = (13, 5)$ which potentially improves the network goodput by 55%. The variations of the goodput with buffer size is depicted in Figure 6.2 to demonstrate the fact that a global optimum exists for this example as well.

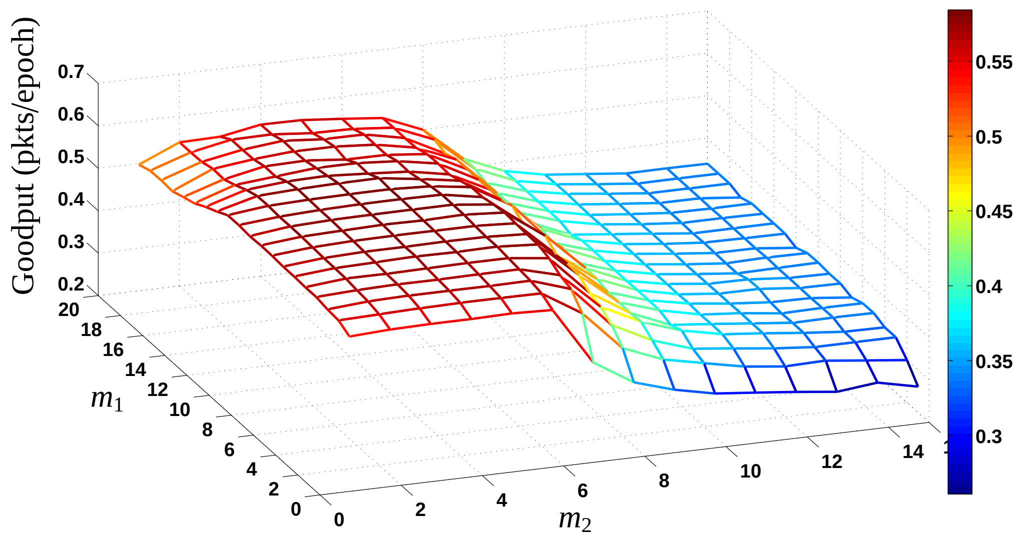

Figure 6.2: Maximized delay-constrained network goodput for varying buffer sizes.

CHAPTER 7

DELAY ANALYSIS OF BURSTY TRAFFIC IN DISRUPTION-TOLERANT NETWORKS

In this chapter, we study sparse mobile ad-hoc networks (*i.e.,* disruption-tolerant networks or DTNs). Our goal is to analytically find the packet latency in such networks for a two-hop unicast scenario with bursty packet arrivals at the source. Similar to the previous chapters, we assume that the intermediate nodes have finite buffers. We exploit an embedded Markov chain approach combined with our proposed iterative estimation technique to study both network delay and queuing delay.

7.1 Introduction and Motivation

Disruption-tolerant networks (DTNs), also referred to as delay-tolerant networks, are a special type of mobile ad-hoc networks. They are often used when there is no backbone infrastructure and hence have applications in military networks, vehicular networks, and providing basic network services to rural areas.

Conventional mobile ad-hoc Networks (MANETs) rely on the existence of end-to-end paths between source and destination regardless of node mobility. However, simultaneous end-to-end connectivity is very rare in DTNs because of the sparseness of nodes in the network. Hence, communication protocols designed for MANETs are unable to perform efficiently for DTNs. Most of the efficient DTN-based schemes [81, 82], use the “store, carry, and forward” paradigm for message delivery, wherein a source node opportunistically transmits packets upon contacting any other node, and relies on the mobility of these “relay” nodes to deliver the message to a certain destination.

Analytical performance modeling for delay-tolerant networks has recently drawn a considerable amount of attention [83–88]. In many cases, the performance of DTNs have been modeled using Poisson process approximations [85–87]. Investigated in [89], a major drawback of this approximation is that assuming Poisson process for contact times does not incorporate the spatial-temporal dependence between contact times of any pair of nodes which is not a realistic assumption in general. Inspired by such shortcomings of the previous works, in [89], Subramanian *et al.* proposed a generalized framework for throughput analysis of finite-buffer delay-tolerant networks. The framework uses the embedded Markov chain approach using which the throughput of such networks can be identified by computing certain well-defined characteristic parameters from the mobility model. Further, the problem of throughput analysis in DTNs has been considered for many different communication scenarios and mobility models in [89–91], and hence, is well-motivated. Although such a framework is useful and valuable for throughput analysis, it is insufficient for modeling the latency performance of DTNs under different types of source-traffic, for the following reasons:

- In order to compute the throughput in the previous model, the source is assumed to be constantly backlogged, *i.e.*, it has infinite number of information packets. Hence, the relay nodes tend to be as congested as possible. Thus, having such an assumption for the source will lead to computing the maximum average "network delay" only.

- The fact that the source is constantly backlogged will naturally eliminate the necessity of defining queueing delay at the source which is an important performance parameter itself.

- The problem of performance analysis of multiple unicast sessions [90] can be useful only when different sources could have different traffic characteristics. In other words, resource sharing protocols will not have a great impact on the performance of the network if all the flows are backlogged at the source and have the same share from the network resources such as buffer space and bandwidth.

Here, as an initial step towards addressing such shortcomings of the previous work [89], we consider the problem of delay analysis for a single unicast session, where a single source node attempts to transmit packets to a single destination using mobile relays. To do so, as our main contribution, a dynamic queue is assumed for the source node with exogenous bursty packet arrivals. By incorporating this seemingly simple addition to the previous problem setting, the new problem turns out to be challenging as we will see in Section 7.3. We will use analytical tools such as embedded Markov chain and Chain-collapsing idea combined with our proposed iterative estimation technique of Chapter 2 to estimate the steady-state distributions of buffer occupancies for relays and the source. We then use these buffer occupancy distributions to obtain analytical expressions for the average delay of packets in a DTN with a general mobility model. Finally, the analytical results are validated using simulations for certain well-known mobility paradigms such as random walk on a grid and random waypoint mobility.

7.2 Network Model

The following setup is considered: n identical nodes, referred to as "relay" nodes, and two other nodes, referred to as "source" and "destination" nodes, are located randomly in a field and moving independently according to a certain mobility model. The relay nodes have the same buffer size of B packets where each packet have a fixed length. However, source and destination nodes have unlimited storage capacity. A discrete-time model is used where at each time epoch, only one packet may be transmitted/received by any node. Further, it is assumed that communication is error-free. Analysis of the problem in presence of channel erasure can be shown to be a straightforward extension of the current framework and will not be discussed here.

7.2.1 Bursty Packet Arrivals at Source

It is known that traffic in communication networks introduces correlations [92, 93]. Here, we use a Bernoulli bursty packet arrival process [94] to model such correlations. Packets are generated according to the model depicted in Fig. 7.1. To be precise, the source alter-

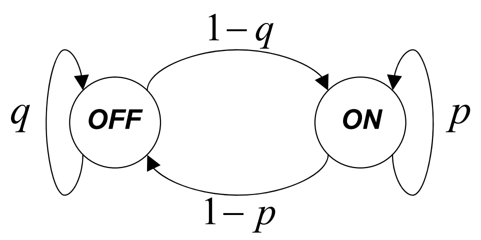

Figure 7.1: Bernoulli bursty arrival model

nates between on-periods, during which exactly one packet is generated per time epoch, and off-period, during which no packets are generated. If the source is "On" or "Off", then it remains in the same state with probability p or q, respectively. At each time epoch, the generated packets are stored at source in a buffer with infinite storage capacity, and are served on a first-come first-served basis. Intuitively, it is more convenient to use mean steady-state arrival rate λ (Packets per epoch) and burstiness factor F instead of the parameters p and q. Given λ, the burstiness factor F takes values between $max\{\lambda, 1-\lambda\}$ and infinity and is a measure for the absolute lengths of on/off periods. The burstiness factor of $F = 1$ represents uncorrelated arrivals which is basically a simple Bernoulli arrival model. The parameters λ and F are derived from the following equations [94]

$$\lambda = \frac{1-q}{2-p-q}, \quad F = \frac{1}{2-p-q}.$$

Here, we only consider mean arrival rates λ which are less than the maximum throughput of the network[1], meaning that the queue at the source remains bounded with high probability and hence, the network is stable[2]. This guarantees the boundedness of the average queueing delay at the source. Note that, by choosing λ above the throughput rate, the queue at the source will grow unboundedly since the network could not deliver packets with such a rate. Thus, without loss of generality, we assume that λ is smaller than the throughput obtained in [89].

[1]We define the maximum throughput as the average number of packets delivered to the destination in each time epoch when the network operates at steady state.

[2]Note that, the queues at relays cannot grow to infinity since they have a finite buffer size.

7.2.2 Interference Model

We assume the communication between a pair of nodes is possible only if they are within the communication range of each other. All the other nodes within the communication ranges of the busy pair are assumed to be silent for the duration of the communication which is one epoch in our problem setup. This is to ensure that there is no wireless interference issues such as hidden-terminal and exposed-terminal situations. Moreover, the source/destination node tries to establish a new link at each epoch, for which several relay nodes may contend. In each time epoch, if the source and destination are within the communication range of each other, then they will form a link, otherwise, if the source or destination are within the communication range of multiple relays, a random relay is selected to setup a link with source or destination, respectively. We say that a "contact" occurs between two nodes whenever they are within the communication range of each other, though they may not communicate. If a pair of nodes win the channel contention, we say that a "link" is established between the communicating nodes.

7.2.3 Routing Protocol

Here, we use a two-hop single-copy routing scheme, meaning, whenever a relay node with available space in its buffer establishes a link with the source, it accepts a packet if the source has any packets available in its queue,*i.e.* it is non-empty, and retains the packet until a link is established with the destination. Packets are served on a first-come first-served basis and no relay-to-relay communication occurs. In addition, the source and destination may, though very rarely, establish a direct link.

7.2.4 Mobility Models

Our framework of analysis is designed to perform well for any mobility model which has stationary properties. This would apply to many well-known models such as random walk on a grid, random waypoint, Brownian motion etc., commonly used in mobile ad-hoc networks research. We assume that each node moves according to the particular chosen mobility model independent of the other nodes in the network. Let $\mathcal{S}_{mob}$ be the set of all states possible in the mobility model. Each "state" of mobility may correspond to information regarding position, direction, velocity, etc. depending upon the underlying mobility model. Let $\chi(t) \in \mathcal{S}_{mob}$ be the state of a single node at any time. It is important to mention that $\chi(t)$ has enough information to determine the probability distribution of $\chi(t+1)$, the state at the next time-step. Typically, one can describe the state transitions for the mobility model by means of a transition function $\boldsymbol{\Psi_{mob}}(\cdot)$ as follows. Let $\mathbf{p}(t)$ be the probability distribution of a node's mobility state at time t. Then, $\mathbf{p}(t+1) = \boldsymbol{\Psi}_{mob}\left[\mathbf{p}(t)\right]$. The transition function $\boldsymbol{\Psi_{mob}}$ depends on the mobility model. Since the mobility model is assumed to be stationary, it has a steady-state probability distribution, $\boldsymbol{\pi}_{mob}$, which satisfies $\boldsymbol{\pi}_{mob} = \boldsymbol{\Psi}_{mob}\left[\boldsymbol{\pi}_{mob}\right]$.

7.3 Markov Chain Analysis

In a DTN with n relay nodes and a single source destination pair, the *state of the network* is defined as the $(2n+3)$-tuple

$$\mathcal{X}(t) = \left(\chi_1(t), \cdots, \chi_n(t), \varphi_1(t), \cdots, \varphi_n(t), \chi_s, \varphi_s(t), \chi_d, \right),$$

where $\chi_k \in \mathcal{S}_{mob}$ is the component describing current mobility state of node k at time epoch t. Also, χ_s and χ_d are the physical mobility states of the source and the destination. The component $\varphi_k(t)$ denotes the buffer occupancy of node k (in packets) at time epoch t. Hence, $0 \le \varphi_k(t) \le B$ at any time t for any node k. Also, $\varphi_s(t)$ denotes the buffer occupancy of the source at time epoch t, where $0 \le \varphi_s(t) < \infty$. Clearly, this describes the state of the network completely: The probabilities of transitions of $\mathcal{X}(t)$ within its state-space can be determined from the mobility model and the communication protocols described previously in Section 7.2. Assuming that the mobility model exhibits stationarity, $\mathcal{X}(t)$ also has a steady state.

The network goes through states wherein packets arrive at the source node, or wherein packets are picked up from the source nodes, or wherein packets are delivered to the destination nodes; these are designated as *active states* for our purpose of delay analysis. Hence, it is sufficient to obtain the steady-state distribution of the entire system described by the state variable $\mathcal{X}(t)$. Steady-state analysis of the network is employed since we are interested in the behavior of the network in the long run. Clearly, the full state-space description of the network is very large to work with. However, we will use the idea of chain-collapsing in the following section to considerably reduce the state-space of the network. Further, for any $x \in [0, 1]$, we define $\overline{x} \triangleq 1 - x$.

7.3.1 The Idea of Chain Collapsing

The full state-space description of the network described above is prohibitively large to work with. In order to reduce the state-space and simplify the analysis, we use the idea of chain-collapsing as in [89]. As the first step, we may try to identify certain symmetries in the network that simplifies the state space. For example, in a scenario where relay nodes are identical, one can view the state of the network from a single relay's perspective. However, the state-space is still very large. Note that, by claiming the full state-space description of the network to be very large, we temporarily ignore the state element corresponding to the buffer occupancy of the source ($0 \leq \varphi_s(t) < \infty$) which is of infinite size. Later, we will observe the challenges of such an extension and will introduce our innovative iterative algorithm to resolve this issue. As the next step, to reduce the state-space further, one can derive a Markov chain from the original state-space such that the steady-state probability distributions are preserved. The above discussion about reducing subsets of states into individual states is thoroughly described in the following theorem from [89]:

Theorem 7.1 *(Chain Collapsing) Let $\mathcal{M}$ be a Markov chain with a set of states denoted by A, with a steady-state distribution $\boldsymbol{\pi}$ for its states. For each $a \in A$, $\pi(a)$ corresponds to the steady-state probability of state a. Let $\{A_i\}_{i=1}^{z}$ be disjoint subsets of A such that $\bigcup_{i=1}^{z} A_i = A$. Then, a new Markov chain defined with $i = 1, \ldots, z$ corresponding to each* *of the above subsets, with transition probabilities corresponding to the "subset-averaged" values of those from the original Markov chain $\mathcal{M}$, has a steady state distribution $\boldsymbol{\pi}' = [\pi(1)\pi(2)\cdots\pi(z)]$ such that $\pi'(A_i) = \sum_{a_j \in A_i} \pi(a_j)$. Moreover, the transition probabilities for the new chain are given by the following relationship:*

$$p'_{A_r A_l} = \sum_{j \in A_r} \sum_{k \in A_l} p_{jk} \pi(j|A_r) = \frac{1}{\pi'(A_r)} \sum_{j \in A_r} \sum_{k \in A_l} p_{jk} \pi(j)$$

Hence, for a particular relay node, we identify all the "desirable" states which contribute to the time packets spend inside the relays and the source, together with certain additional "auxiliary" states to arrive at an "embedded" Markov chain. The idea of chain-collapsing enables us to extract only the necessary information from the original Markov chain. In particular, the performance computation problem is reduced to computing the steady-state probabilities of certain subsets of a well-defined embedded Markov chain. Note that, we are not interested to find individual steady-state probabilities of states within one particular desired subset. The rest of the analysis involves the computation of the transition probabilities between the desired subsets followed by computation of their steady-state probabilities using the collapsed chain.

7.3.2 Embedded Markov Chain for a Relay Node

Here, our main goal is to define the desired states of the embedded Markov chain for a single relay node so that the resulting steady-state probabilities could provide us with sufficient information to approach the problem of delay analysis. Hence, we define the embedded Markov chain for a relay node according to the following subsets of states:

- Let S_i $(1 \leq i \leq B)$ be the set of network states wherein the most recent link that node v had was with a non-empty source, resulting in i packets in the buffer after receiving a packet.

- Similarly, let D_j $(0 \leq j \leq B-1)$ be the set of network states wherein the most recent link that node v had was with the destination, resulting in j packets in its buffer after

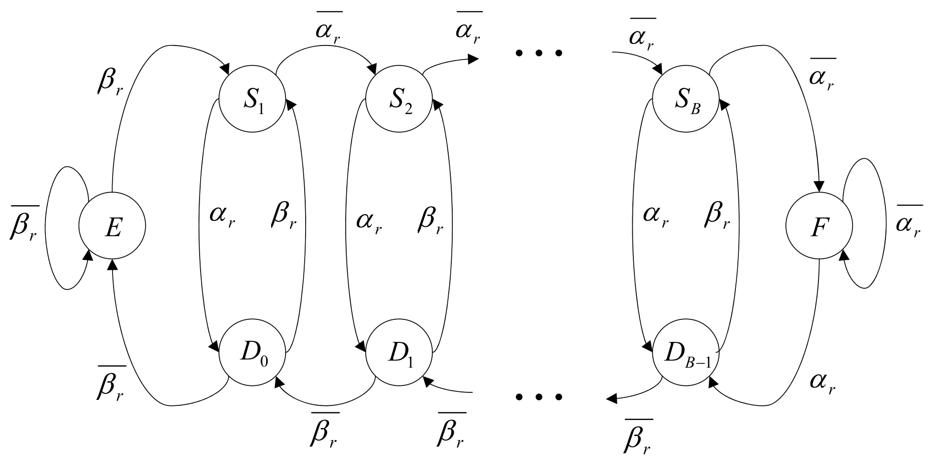

Figure 7.2: The embedded Markov chain for a relay node (RMC)

transmitting a packet.

- Let F be the set of network states wherein the most recent link that node v had was with a non-empty source, but v was unable to accept any packet due to lack of buffer space (*i.e.*, Full buffer state).

- Similarly, let E be the set of network states wherein the most recent link that node v had was with the destination, but v had no packet to transmit (*i.e.*, Empty buffer state).

Given the state transition probabilities for the embedded Markov chain in Fig. 7.2 (RMC), a closed-form expression for its steady-state probabilities can be easily obtained using

$$\begin{aligned}\Pr\{F\} &= \left(\frac{\overline{\alpha_r}}{\overline{\beta_r}}\right)^B \frac{\beta_r}{\alpha_r}\Pr\{E\},\\ \Pr\{S_{i+1}\} = \Pr\{D_i\} &= \left(\frac{\overline{\alpha_r}}{\overline{\beta_r}}\right)^i \frac{\beta_r}{\overline{\beta_r}}\Pr\{E\},\end{aligned}$$

for $i = 0, \ldots, B-1$, and,

$$\Pr\{E\} = \begin{cases} \frac{1}{2}\left\{1 + \frac{\beta_r}{\overline{\beta_r}}B\right\}^{-1}, & \text{if } \alpha_r = \beta_r \\ \frac{\alpha_r - \beta_r}{\alpha_r + \beta_r}\left\{1 - \frac{\beta_r}{\alpha_r}\left(\frac{\overline{\alpha_r}}{\overline{\beta_r}}\right)^B\right\}^{-1}, & \text{if } \alpha_r \neq \beta_r \end{cases}.$$

Further, the state transition probabilities for RMC, *i.e.*, α_r and β_r, can be obtained using the following lemma.

Lemma 7.1 *Let α_0 be the probability that a node currently in contact with the source (or destination) will have a contact with the destination (or source) before coming in contact with the former again. Also, let p_c be the probability that a relay node loses contention on meeting the source/destination node. Finally, let p_e be the probability that source node is empty,* i.e. *has no packets in its queue, when meeting a relay or destination node. Then,*

$$\alpha_r = \frac{\alpha_0}{\overline{p_e}(2\alpha_0 p_c + \overline{p_c}) + \alpha_0 p_e}, \quad \beta_r = \overline{p_e}\alpha_r.$$

The proof is very similar to the proof of Lemma 7.4 which can be found in section A.1 of Appendix A.

The parameter α_0 in Lemma 7.1 is characterized for a general mobility model in the following lemma.

Lemma 7.2 *Let T_0 be a random variable representing the inter-contact duration of two nodes, and let T_∞ be the random variable representing the waiting time until two nodes meet, given that they are distributed according to the steady-state spatial location distribution. Then we have*

$$\alpha_0 = \sum_{\tau=1}^{\infty} F_{T_\infty}(\tau) P_{T_0}(\tau),$$

where $P_{T_0}(\tau)$ and $F_{T_\infty}(\tau)$ are the probability density function of T_0 and the cumulative density function of T_∞, respectively.

The proof is very similar to the proof of Lemma 7.6 which can be found in section A.2 of Appendix A.

Finally, since the contention failure probability p_c in Lemma 7.1 only depends on the mobility and routing protocol, hence the result from [89] can be exploited to derive p_c using the following lemma.

Lemma 7.3 *Let $\mathbf{x}'$ be the subset of states wherein a contact with a node in state $\mathbf{x} \in \mathcal{S}_{mob}$ can be established and $\boldsymbol{\pi}_{spt}$ be the steady-state spatial node-location distribution due to the* *underlying mobility model.*

$$p_c = 1 - \left(1 - E[T_0]^{-1}\right) \sum_{k=0}^{n-1} \sum_{\mathbf{x}} \tfrac{\pi_{spt}(\mathbf{x})}{k+1} \tbinom{n-1}{k} \pi_{spt}^{k}(\mathbf{x}') \times \left\{1 - \pi_{spt}(\mathbf{x}')\right\}^{n-1-k}$$

7.3.3 Embedded Markov Chain for the Source

Here, we define the desired states of the embedded Markov chain for the source node so that the resulting steady-state probabilities could be helpful for the problem of delay analysis. Hence, the embedded Markov chain for the source node is defined according to the following subsets of states:

- Let A_i ($i = 1, 2, \ldots$) be the set of network states wherein the most recent event at the source is a packet arrival (on-period) resulting in i packets in the source queue.
- Also, let R_j ($j = 0, 1, \ldots$) be the set of network states wherein the most recent event at the source is meeting a non-full relay or the destination during an off-period, resulting in j packets in the source buffer.
- Finally, let E be the set of network states wherein the most recent event at the source is meeting a non-full relay or the destination during an off-period while the source node is empty and hence no packet is transmitted.

Given the state transition probabilities for the embedded Markov chain in Fig. 7.3 (SMC), a closed-form expression for its steady-state probabilities can be easily obtained using

$$\begin{aligned} \Pr\{R_i\} = \overline{\gamma_s}\,\Pr\{A_{i+1}\} &= \left(\frac{\overline{\gamma_s + \alpha_s}}{\overline{\beta_s}\,\overline{\gamma_s}}\right)^{i} \frac{\beta_s}{\overline{\beta_s}} \Pr\{E\}, \quad i \geq 0 \\ \Pr\{E\} &= \frac{\alpha_s - \overline{\gamma_s}\beta_s}{\alpha_s + \beta_s}. \end{aligned}$$

Moreover, the state transition probabilities of SMC, *i.e.*, α_s, β_s and γ_s, can be derived using the following lemmas.

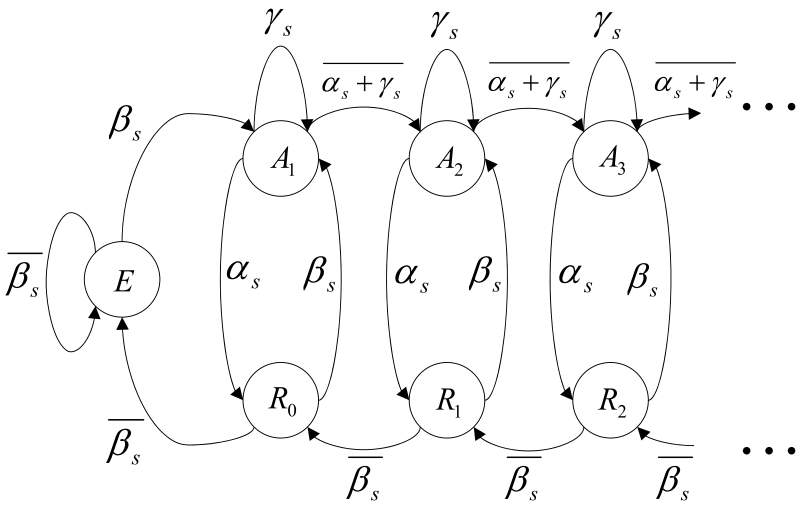

Figure 7.3: The embedded Markov chain for the source node (SMC)

Lemma 7.4 *Let α_1 be the probability that the source node in its on-period, will have a contact with any other node (relay or destination) before another packet arrives. Further, let α_2 be the probability that the source node, currently in contact with a node (relay or destination), will have an arriving packet before coming in contact with any other node. Finally, let p_f be the probability that a relay node is full when meeting with the source and is unable to accept any packets. Then, we have*

$$\alpha_s = \frac{\alpha_1 \overline{p_b}}{\alpha_2 p_b + \overline{p_b}}, \quad \beta_s = \frac{\alpha_2}{\alpha_2 p_b + \overline{p_b}},$$

where, $p_b = \frac{n}{n+1} p_f$.

The proof can be found in section A.1 of Appendix A.

Lemma 7.5 *Let β_1 be the probability that given the source node has no contacts with any other node (relay or destination), it will contact a node during the next time epoch. Further, let β_2 be the probability that the source node, currently in contact with a node (relay or destination), will have contact with none of the other nodes during the next time epoch. Then, we have*

$$\gamma_s = \frac{\beta_1 \overline{p_b}}{\beta_2 p_b + \overline{p_b}},$$

where, p_b is as defined in Lemma 7.4.

The proof is very similar to the proof of Lemma 7.4 which can be found in section A.1 of Appendix A..

Finally, the parameters α_1, α_2, β_1 and β_2, can be characterized for a general mobility model and source traffic model using the following lemma.

Lemma 7.6 *Let S_0 be a random variable representing the inter-arrival duration of packets arriving at source, and let S_∞ be the random variable representing the waiting time until an arrival given the source is currently in off-period. Further, let $T_{\infty,n}$ be a random variable representing the waiting time until a contact with at least one of the $n+1$ relays/destination occur for the source, given that the nodes are distributed according to the steady-state spatial location distribution. Also, let $T_{0,n}$ be a random variable representing the waiting time until source makes contacts with at least one of the $n+1$ relays/destination nodes, given that the source is currently in contact with a relay/destination and the other n nodes are distributed according to the steady-state spatial location distribution. Then, we have*

$$\begin{aligned}
\alpha_1 &= \sum_{\tau=1}^{\infty} F_{T_{\infty,n}}(\tau) P_{S_0}(\tau), \\
\alpha_2 &= \sum_{\tau=1}^{\infty} \left(1 - F_{T_{0,n}}(\tau)\right) P_{S_\infty}(\tau), \\
\beta_1 &= \Pr\{T_{\infty,n} = 1\}, \\
\beta_2 &= 1 - \Pr\{T_{0,n} = 1\}.
\end{aligned}$$

The proof can be found in section A.2 of Appendix A.

7.3.4 Iterative Estimation

Thus far, we have developed two different collapsed Markov chains RMC and SMC originated from the full state-space of the entire network. In other words, we have observed the desired states in the network from the point of view of a single relay node and the source node. However, it is notable that deriving the state transition probabilities for RMC and SMC requires using Lemmas 7.1, 7.4 and 7.5 in which the parameters p_f and p_e are not known in advance. In this section, we will see that these two Markov chains are not only dependent on each other but also closely related. Further, their dependency could lead us into solving both of them using an iterative algorithm.

We start from the problem of finding the probability p_f. We need to know the portion of relay-source links during which a relay is full. Using steady-state probabilities of RMC, we have the following

$$p_f = \overline{\alpha_0}\ \overline{p_c}\ \frac{\Pr\{F\} + \Pr\{S_B\}}{\Pr\{F\} + \sum_{i=1}^{B} \Pr\{S_i\}}. \tag{7.1}$$

Further, obtaining the steady-state probabilities of RMC requires having its state transition probabilities by using Lemma 7.1. Hence, we need to find the probability p_e which is the portion of source-relay/destination links during which the source is empty. Using steady-state probabilities of SMC, the following relation can be obtained

$$p_e = \overline{\alpha_2 + \beta_2}\ \frac{\Pr\{E\} + \Pr\{R_0\}}{\Pr\{E\} + \sum_{i=0}^{\infty} \Pr\{R_i\}}. \tag{7.2}$$

Finally, obtaining the steady-state probabilities of SMC requires having its state transition probabilities by using Lemmas 7.4, 7.5 and consequently, knowing p_f. Interestingly, we are back to where we started. This hints us that the problem might tend to have an iterative solution. In [31], we developed an iterative algorithm to estimate the capacity of finite-buffer line networks (non-mobile). Likewise, here we propose an iterative estimation algorithm to estimate the unknown parameters stated in the discussion above, starting from some initial values, *e.g.*, $p_f = 0$. The schematic in Fig. 7.4 shows the iteration steps. The iteration procedure will go on until convergence of the steady-state probability vectors. One way to measure the convergence of our method is to compare the Euclidean distance between the vectors of each two consecutive iterations and stop the procedure when the distance becomes smaller than a previously chosen threshold ϵ.

7.3.5 Delay Analysis

Using the iterative estimation technique of Section 7.3.4, the estimated steady-state probabilities for RMC and SMC are obtained. In this section, we use such results to find analytical expressions for the average packet delay in DTNs.

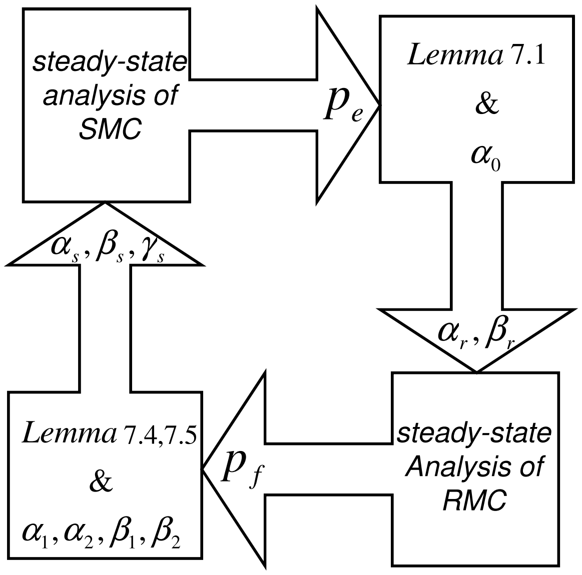
steady-state analysis of SMC
p_e
$Lemma$ 7.1 & α_0
α_r, β_r
steady-state Analysis of RMC
p_f
$Lemma$ 7.4,7.5 & $\alpha_1, \alpha_2, \beta_1, \beta_2$
$\alpha_s, \beta_s, \gamma_s$

Figure 7.4: A graphical presentation of the iterative estimation algorithm

We divide the latency experienced by each packet to two parts: "*Network Delay*" and "*Queueing Delay*". The network delay is defined as the total time spent by a packet inside the buffer of a relay node which is the time it takes from the instant when the packet leaves the source node until when it reaches the destination node. The queueing delay is defined as the time spent by a packet inside the queue of the source node which is the time it takes from the instant when the packet arrives at the source node until successfully leaving it. The analytical expressions for both average network delay and average queueing delay at the source are obtained by using the following propositions. The total average packet latency can be simply derived by adding both the average network delay and the average queueing delay.

Proposition 7.1 *Let P_z be the portion of the packets that experience zero network delay due to the event that a direct link between the source and the destination is established. Given* $\Pr\{S_i\}$ *for $i = 1, 2, \ldots, B$ from the steady-state analysis of RMC, the average network* *delay can be obtained from*

$$D_{net} = \overline{P_z} \sum_{i=1}^{B} \frac{\Pr\{S_i\}}{\sum_{j=1}^{B} \Pr\{S_j\}} \left(E[T_\infty] + (i-1)E[T_0]\right),$$

where $P_z = \frac{(n+1)E[T_{0,n}]}{n\overline{p_c}E[T_0]}$*, and the contention failure probability* p_c *is derived in Lemma 7.3.*

The proof can be found in section A.3 of Appendix A.

Proposition 7.2 *Given* $\Pr\{A_i\}$ *for* $i = 1, 2, \ldots$ *from the steady-state analysis of SMC, the average queueing delay at the source can be derived using*

$$D_{queue} = \overline{p_b}^{-1} \sum_{i=1}^{\infty} \frac{\Pr\{A_i\}}{\sum_{j=1}^{\infty} \Pr\{A_j\}} \left(E[T_{\infty,n}] + (i-1)E[T_{0,n}]\right),$$

where the blocking probability p_b *is defined in Lemma 7.4.*

The proof can be found in section A.4 of Appendix A.

7.4 Simulation Results

In this section, we present the simulation results for validation of our analytical framework. Our analytical results are compared to the simulations of sparse mobile ad-hoc networks for two well-known mobility models.

7.4.1 Random Walk on a Grid Mobility Model

In this model, nodes are randomly moving on a $M \times M$ square grid as shown in Fig. 7.5. At each time epoch, nodes may remain in the same cell in the grid, or move to an adjacent cell in the next time step with a certain probability. The transition probabilities for the random walk are chosen so that it results in a uniform steady-state spatial distribution, *i.e.*, a node is located in a specific cell with probability $\frac{1}{M^2}$. Hence, the probability of transition to adjacent cells is $\frac{1}{5}$ and the self-transition probability for each cell will be $1 - \frac{\text{No. of adj. cells}}{5}$. As an example, for the cell in the corner, the self-transition probability is equal to $\frac{3}{5}$. The contention failure probability p_c can be derived using Lemma 7.3 from the

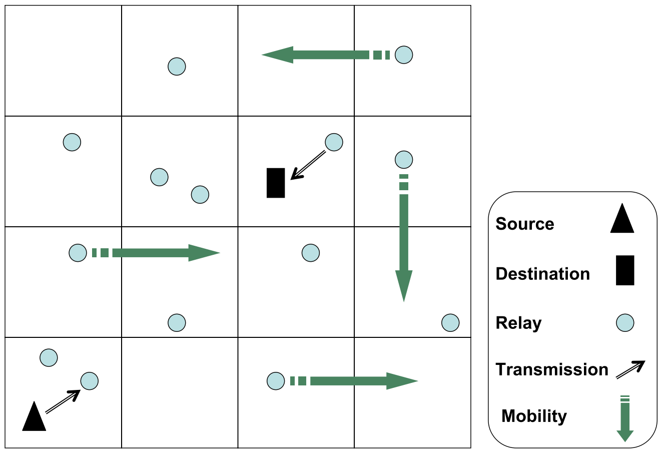

Figure 7.5: Network Model

following relation

$$p_c = 1 - \frac{M^2}{n}\left(1 - \left(1 - \frac{1}{M^2}\right)^n\right).$$

The mobility parameters needed for Lemmas 7.1, 7.4, 7.5 can be obtained as well. In [89], an analytical approximation is proposed to find such parameters for the case of random-walk on a grid.

Here, we choose the node buffer size to be 10 packets, while the number of relay nodes is kept at 10 and the grid size is 8×8. Fig. 7.6 and Fig. 7.7 demonstrate the accuracy of our estimation for average queueing delay at the source and average network delay, respectively. Further, variations of both queueing delay and network delay with mean arrival rate λ and burstiness factor F are presented. As stated before, validation of our iterative estimation algorithm is performed for arrival rates λ smaller than the throughput of the network. By increasing λ to the values close to the network throughput, the average queueing delay at the source goes to infinity. However, average network delay will remain bounded from above since all the relays have finite buffer size. In other words, by approaching more and more to the network throughput, queueing delay at the source becomes the dominant term comparing to the network delay. Finally, it can be observed that, higher burstieness factor results in larger latencies for packets inside the queue of the source.

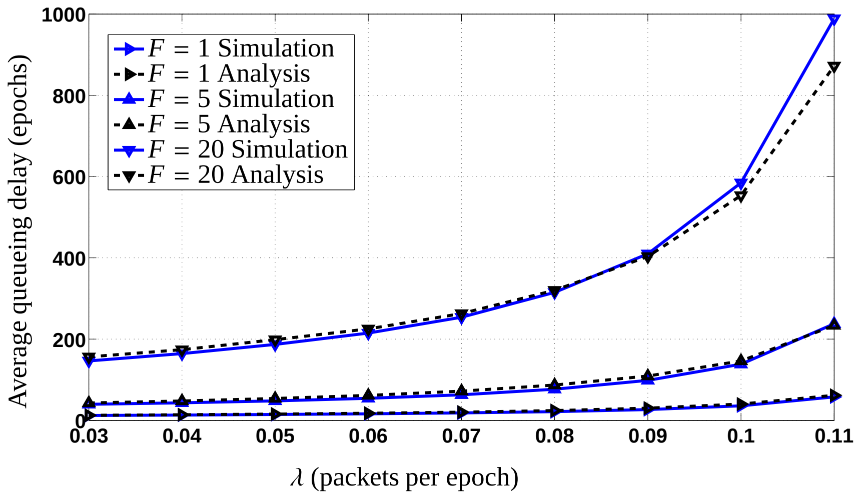

Figure 7.6: Variations of average queueing delay at the source with mean arrival rate λ and burstieness factor F for a random walk on a grid mobility model

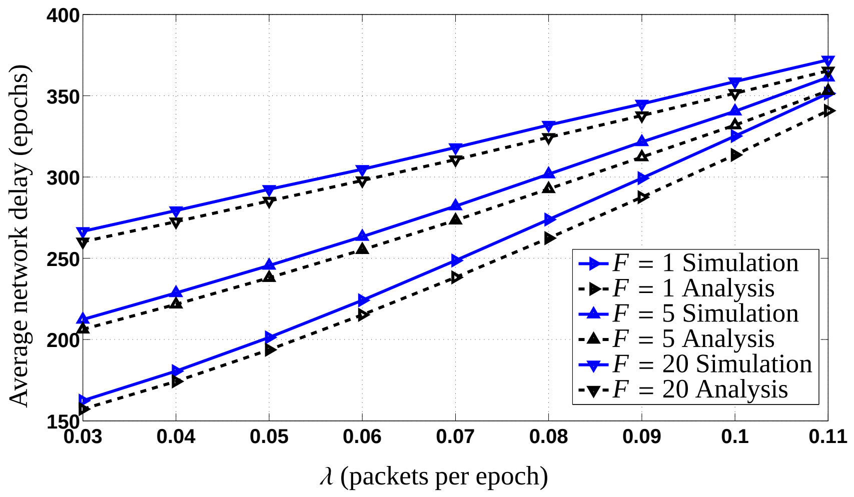

Figure 7.7: Variations of average network delay with mean arrival rate λ and burstieness factor F for a random walk on a grid mobility model

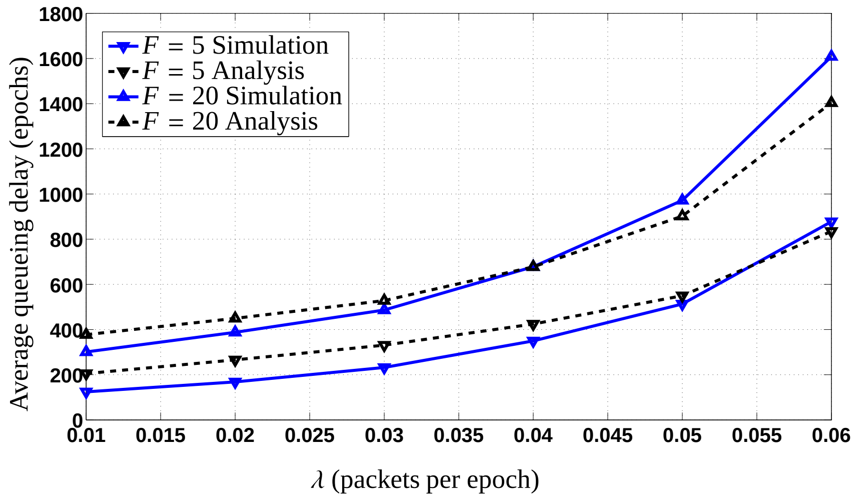

Figure 7.8: Variations of average queueing delay at the source with mean arrival rate λ and burstieness factor F for a random waypoint mobility model

7.4.2 Random Waypoint Mobility Model

The random waypoint mobility model is commonly used in simulation studies of networking protocols. Here, each node selects a random location in the deployment area, and moves towards that location with a random speed. Upon reaching to its target location, it waits for a random amount of time, and then the next location and speed are chosen. The mobility parameters needed for Lemmas 7.1, 7.4, 7.5 have not been obtained in closed form in the literature, to the best of our knowledge. However, some approximations [89] are available for the steady-state spatial node distribution, and can be used to compute the contention failure probability p_c. Here, we have obtained the mentioned mobility parameters numerically by a quick simulation of the mobility only. The deployment area is chosen to be a $5km \times 5km$ square region where 10 nodes are deployed in random locations. The node velocity is chosen from a uniform distribution with $V_{min} = 3m/s$ and $V_{max} = 30m/s$. The communication range is chosen to be $500m$. The pause-time is also modeled as an exponential distribution with a mean of $20s$. Finally, the node buffer size is chosen to be 10 packets.

Fig. 7.8 and Fig. 7.9 demonstrate the accuracy of our estimation for average queueing delay at the source and average network delay, respectively. Here, for clarity of the

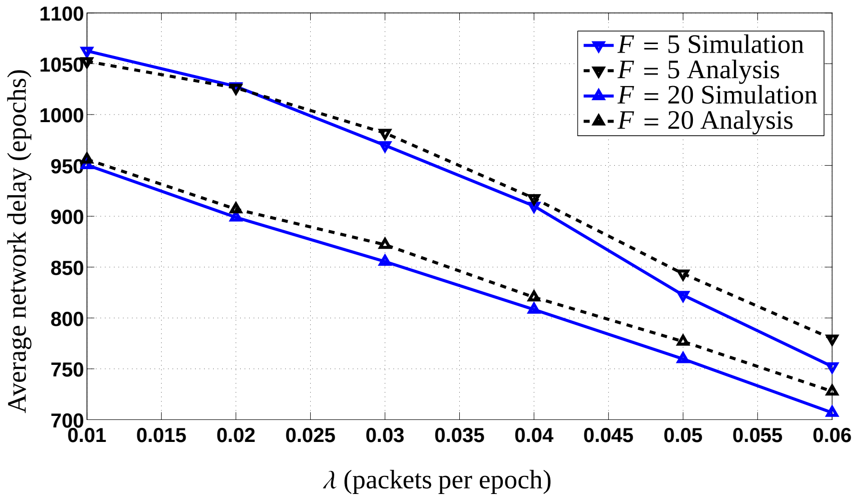

Figure 7.9: Variations of average network delay with mean arrival rate λ and burstieness factor F for a random waypoint mobility model

presentation, we only demonstrate the average delay variations for the cases $F = 5$ and $F = 20$ since the curves were close together. It is interesting to observe that, in Fig. 7.9, by increasing the mean arrival rate λ the average network delay decreases. The reason for such a behavior is solely contributed to the slow nature of the specific mobility parameters (due to lower speed comparing to the area and the waiting periods). As an example, for the random waypoint mobility, the quantity $E[T_{\infty,n}]$ is about 30 times larger than $E[T_{0,n}]$, where the former is only about 2 times larger than the latter for the case of random walk on a grid mobility model. This means that when the mean arrival rate increases, many packets will be trapped inside the relays and take too much time to be released while keeping the relays full. Meanwhile, the proportion of packets with zero network delay will increase as the proportion of the packets transferring directly from source to the destination increases. However, this comes with a cost which would be a huge increase in average queueing delay as it can be observed in Fig. 7.8.

CHAPTER 8

THROUGHPUT AND DELAY OF BLOCK-BASED RANDOM LINEAR CODING IN LINE NETWORKS

In this chapter, a block-by-block random linear network coding (RLC) scheme with feedback on the links is selected for reliability and more importantly guaranteed decoding of each block in a certain time. We use our proposed iterative estimation algorithm to find the performance parameters of the network and more importantly reduces the computational complexity compared to the exact analysis. We will see that the proposed framework yields an accurate estimate of the distribution of buffer occupancies using which we obtain analytical expressions for network throughput and delay distribution of a block of packets.

The RLC scheme for finite-buffer networks introduced in [28] has the limitation that it cannot be used to characterize the latency profile. This is because, the typical notion of latency is not meaningful for the RLC scheme. Since latency is critical for real-time applications (such as video streaming), a block-by-block encoding of the stream is required. Hence, we introduce our *Block-based Random Linear Coding* scheme, which applies RLC on each individual block[1]. We will see that our approach guarantees a decoding delay within a certain amount of time while this is not the case in general for RLC which is a rate-optimal scheme with potentially a large decoding delay as we will see in Chapter 10. Further, by using out proposed block-by-block RLC scheme, only one feedback is required for transmission of K packets which considerably limits the average number of feedbacks per transmission.

8.1 Network Model and Coding Scheme

We consider a line network of h hops with the vertex set $V = \{v_0, v_1, \ldots, v_h\}$ and the edge set $\overrightarrow{E} = \{(v_i, v_{i+1}) : i = 0, \ldots, h-1\}$ for some integer $h \geq 2$. Let ε_i denote the packet

[1]Just as in any network coding scheme, the packets received by the destination are linear combinations of the original data in a block. Hence, with the knowledge of this linear transformation at the decoder, inversion can be performed to recover the data block.

erasure probability over the link (v_{i-1}, v_i). Each node $v_i \in V$ has a buffer size of m_i packets. It is assumed the destination node has no buffer constraints and that the source node has infinitely many innovative packets[2].

The system is analyzed using a discrete-time model, where each node transmits one packet over a link per epoch. We introduce a practical network coding scheme, which is well-suited for transmitting real-time data streams in a block-by-block fashion using RLC and feedback. In the proposed scheme, the source node takes the stream of packets and divides them into blocks of K packets each. The buffer of each intermediate node $v_i \in V$ is then segmented into M_i blocks. In other words, we have $m_i = M_i K$. Each block is then served using RLC over all the packets in the block. The blocks are served based on a *first-come first-serve* policy. An instant lossless hop-by-hop acknowledgment per block is also employed to indicate the successful receipt of a complete block of K packets. In each epoch, one or multiple of the following events occur in different orders:

1. If a node neither receives any innovative packet nor conveys any innovative packet to the next node, then the content of its buffer does not change[3].

2. Upon receiving an innovative packet, it will be stored in the last available block (a block of memory with less than K innovative packets stored in it). The packets of this block will be served after all the previously received blocks in the buffer are completely served.

3. In each epoch, every node transmits a packet formed by the RLC encoding over its current block (oldest block in the queue), until the node receives an acknowledgment indicating that the block is fully conveyed to the next-hop node. The block will then be removed from the buffer and the next block in the queue will be served. This also implies that free space in the buffer will be increased by K packets.

[2]A received packet is called *innovative* with respect to a node if the packet cannot be generated by a linear combination of the current buffer contents of the node.

[3]Here, by *conveying a packet*, we mean that *the packet is successfully transmitted and stored at the next-hop node*.

To implement the per-block feedback mechanism, a node must distinguish innovative packets upon their reception. One way is to compute the rank of the received packets in a block. A more practical protocol is to use a variable CMB in the header of each encoded packet, indicating the number of innovative packets used by RLC to form the packet. Further, every node maintains a counter INV indicating the number of innovative packets received in its current block so far. Every node sets INV = 0 for each new incoming block[4] and increments it by one for each incoming packet whose CMB is greater than INV of the receiving node. Note that, if the current block in the queue of a node has K innovative packets then CMB is equal to K for all the packets to be transmitted by that node.

We will employ the following notations. For any $x \in [0, 1]$, $\overline{x} \triangleq 1 - x$. The convolution operator is denoted by $\otimes$ and $\otimes^l f$ is used as a shorthand for the l-fold convolution of f with itself. $\mathbb{G}(p)$ denotes the geometric distribution with mean inter-arrival time $\frac{1}{1-p}$.

8.2 Exact analysis and Network States

In [29], a Markov-chain approach for exact analysis of a finite-buffer line network identifies the throughput as equivalent to the problem of finding the buffer occupancy distribution of the intermediate nodes. However, the size of the Exact Markov Chain (EMC) and the multiple reflections due to the finiteness of buffers at each intermediate node render this problem mathematically intractable for even networks of small hop-lengths and buffer sizes. We therefore aim to approximate the distribution of buffer occupancies.

To approach this approximation problem properly, it is necessary to clearly define the buffer states in a manner that (a) an irreducible ergodic Markov chain is obtained, and (b) the steady-state distribution of the chain allows tractable expressions for the performance parameters of the network. Thus, a proper definition for the buffer states cannot be proposed unless the communication scheme is known. For the scheme in Section 8.1, two variables are needed to track all the buffer states of a node. Let s be the total number of innovative

[4]If INV = K, then the counter is reset to 0 and an acknowledgment is sent to the previous node.

(with respect to the next-hop node) packets stored at a node. Denote t to be the number of successfully conveyed innovative packets by the node from the current block. Then, the pair (s, t) can be defined as the state of the buffer of the node. Note that s is the minimum number of packets that a node has to store in its buffer. Also note that, since a new block starts to be served after the K^{th} packet of the current block, $t \in \{0, \ldots, K - 1\}$. As an example, assuming that node v_i is in state (s, t) at the start of the epoch, given that during the epoch it only sends a packet successfully but does not receive any packets, the state of the network will change to $(s, t + 1)$ if $t = \{0, 1, \ldots, K - 2\}$ or it will change to $(s - K, 0)$ if $t = K - 1$.

8.3 Approximate Markov Chain Modeling

In this section, we determine the distribution of buffer occupancy of an intermediate node, which will later be used to analyze network parameters such as throughput and latency of a block.

Due to the discrete-time nature of the analysis framework, two Markov chains need to be constructed for each intermediate node. The first one considers the buffer occupancy at instants when a packet has just been transmitted (either successfully or unsuccessfully), which is called *receive-first Markov chain* (RFMC). This is required to compute the probability of blocking, which is caused when the state of a node is forced to remain unchanged because the transmitted packet was successfully delivered to the next-hop node, but the latter does not store the packet due to full buffer occupancy . The second one considers the buffer occupancy at instants when a packet has just been received/stored, which is called *transmit-first Markov chain* (TFMC). This will be used to calculate the incoming rate of innovative packets at each node.

Note that the problem of exactly identifying the steady-state probabilities of the RFMC and TFMC suffers the same difficulties as identifying that of the EMC [31]. The finite buffer condition introduces a strong dependency of state update at a node on the state of the node that is downstream. To develop an estimation scheme that considers blocking, we make the following assumptions.

1. Packets are ejected from nodes in a memoryless fashion. This assumption allows us to keep track of only the information rate.

2. The blocking event occurs independent of the state of a node. This allows us to track just the blocking probability.

3. At any epoch, given the occupancy of a particular node, packet arrival and blocking events are independent of each other.

These assumptions spread the effect of blocking equally over all non-zero states of occupancy at each node. Now, given that the arrival rate of innovative packets at node v_i is r_i packets/epoch, and that the probability of the next node being full, (i.e., $s = m_{i+1}$ for node v_{i+1}) is $p_{b_{i+1}}$, we can show that transition dynamics of the state change for node v_i is given by the Markov chain depicted in Fig. 8.1 for both TFMC and RFMC [5]. Also, obtaining the

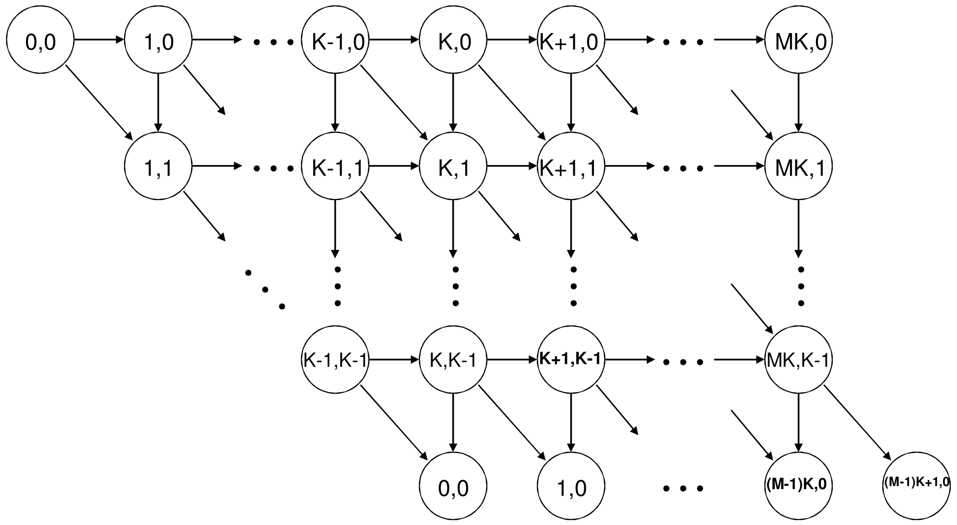

Figure 8.1: The general structure for both TFMC and RFMC for a node with buffer size $m = MK$.

state transition probabilities is straightforward using r_i, $p_{b_{i+1}}$, ε_i and ε_{i+1}. As an example,

[5]Self-loops are not demonstrated in the figure.

for TFMC we have the following

$$P^{TF}_{(s,t)\to(s+1,t)} = \begin{cases} r_i & s = t \\ r_i(\varepsilon_{i+1} + \overline{\varepsilon}_{i+1} p_{b_{i+1}}) & s-1 \geq t \\ 0 & \text{Otherwise} \end{cases} . \tag{8.1}$$

Note that we notate s and t instead of s_i and t_i for the simplicity of notation when considering node v_i. The same transition probability for RFMC is different from TFMC and is given by

$$P^{RF}_{(s,t)\to(s+1,t)} = r_i(\varepsilon_{i+1} + \overline{\varepsilon}_{i+1} p_{b_{i+1}}). \tag{8.2}$$

Note that, both (8.1) and (8.2) are valid for $s = \{0, 1, \dots, M_iK - 1\}$ and $t = \{0, 1, \dots, K - 1\}$.

For all input parameters, the Markov chains can be shown to be aperiodic, irreducible and ergodic. Therefore, it possesses a unique steady-state distribution. The steady state probability of node v_i being in state (s, t), is denoted by $P_i^{RF}(s, t)$ and $P_i^{TF}(s, t)$ for RFMC and TFMC, respectively.

The blocking probability that the node v_{i-1} perceives from the node v_i is the same as the probability of v_i being full ($s = M_iK$) at the instant when a packet is transmitted successfully by the previous node. Hence, this probability have to be calculated using the steady state probability distribution of RFMC as follows

$$p_{b_i} = \sum_{t=0}^{K-1} P_i^{RF}(M_iK, t). \tag{8.3}$$

Similarly, the steady state probability distribution of TFMC can be used to compute the arrival rate at the next node using

$$r_{i+1} = (1 - \sum_{t=0}^{K-1} P_i^{TF}(t, t))\overline{\varepsilon}_{i+1}. \tag{8.4}$$

Note that if a node is in the state (t, t) ($t \in \{0, \dots, K - 1\}$), it means that it has stored t innovative packets from the current block so far and it has also sent t linear combinations of them successfully. Therefore, there is no more innovative packets to send.

The approximate solution is obtained iteratively by the following procedure:

Step 1. Initialization: $\mathbf{r} = (\overline{\varepsilon}_1, \ldots, \overline{\varepsilon}_h)$ and $\mathbf{p_b} = (0, \ldots, 0)$

Step 2. Construct TFMC and RFMC respectively and compute their steady-state probabilities.

Step 3. Using $P_i^{RF}(s,t)$ and $P_i^{TF}(s,t)$ (obtained from step 2) calculate the new values for $\mathbf{r}$ and $\mathbf{p_b}$ by (8.3) and (8.4), respectively, for $i = 1, 2, \ldots, h-1$ and auxiliary equations $r_1 = 1 - \varepsilon_1$ and $p_{bh} = 0$.

Step 4. Repeat Step 2 and Step 3 until all the distributions converge.

8.4 Computation of Network Parameters

In this section, we exploit the results of the iterative estimation of buffer occupancy distributions in Sec. 8.3 to obtain analytical expressions for both network throughput and delay distribution of a block. The *Block Delay* is defined as the time taken for a block of K information packets (at the source) to be transferred through a line network from the instant when the first packet of that block is transmitted from the source node to the instant when the K^{th} innovative packet of that block is received by the destination node (i.e., the block can be decoded at the destination).

Given a line network with link erasures $\mathcal{E} = (\varepsilon_1, \ldots, \varepsilon_h)$, intermediate node buffer sizes $\mathcal{M} = (m_1, \ldots, m_{h-1})$, we can find the approximate solution $(\mathbf{r}, \mathbf{p_b})$. Using this, an estimate of the throughput is obtained using the following

$$\mathcal{C}(\mathcal{E}, \mathcal{M}, K) = r_h(1 - p_{bh}) = r_h.$$

To compute the distribution of the total *block delay*, one can proceed in a hop-by-hop fashion in the following way:

$$\mathbf{D} = \mathbf{T}_1 \otimes \mathbf{W}_1 \otimes \mathbf{W}_2 \otimes \ldots \otimes \mathbf{W}_{h-2} \otimes \mathbf{F}, \tag{8.5}$$

where $\mathbf{T}_1$ is the probability distribution of the time taken for a packet in the source to be conveyed to node v_1. Further, $\mathbf{W}_i$ is the probability distribution of the time taken from the instant when node v_i stores the first innovative packet of a block to the instant the first packet of the corresponding block in v_i is conveyed to node v_{i+1}. Finally $\mathbf{F}$ is the probability distribution of the time taken for all the K packets of a block in node v_{h-1} to be conveyed to the destination node from the instant when the first innovative packet of the same block is stored in the buffer of node v_{h-1}. Thus, using the definition, we have $\mathbf{T}_1 = \mathbb{G}(\acute{\varepsilon}_1)$, where $\acute{\varepsilon}_1$ is the effective erasure probability (after considering blocking) and is given by

$$\acute{\varepsilon}_i = \begin{cases} \varepsilon_i + p_{b_i}\bar{\varepsilon}_i & i = 1, 2, \ldots, h-1 \\ \varepsilon_h & i = h \end{cases}. \tag{8.6}$$

Also, the average waiting time in node v_i is formulated as

$$\mathbf{W}_i = \pi_i(0,0)\mathbf{S}_i(0,0) + \sum_{d=1}^{M_i-1}\sum_{t=0}^{K-1}\pi_i(dK,t)\mathbf{S}_i(dK,t), \tag{8.7}$$

where, $\pi_i(s,t)$ is the probability that an arriving packet finds node v_i in state (s,t) given that it is the first packet of its corresponding block. Also, $\mathbf{S}_i(s,t)$ is the probability distribution of the time taken for the first innovative packet of a block in v_i to be conveyed to node v_{i+1} from the instant when the first innovative packet of that block arrives at node v_i and finds its buffer at state (s,t).

If an arriving packet is the first of its corresponding block, it finds the buffer at states of the form (dK,t) where d can take any value between 0 and $M_i - 1$. This is because of the fact that the last block had been completely served before the first packet of the current block arrives. Hence, both $\mathbf{S}_i(s,t)$ and $\pi_i(s,t)$ will be 0 if s is not a multiple of K. Finally, $\pi_i(s,t)$ can be formulated as follows for $n \in \{0, 1, \ldots, M_i - 1\}$,

$$\pi_i(s,t) = \begin{cases} \frac{P_i^{RF}(s,t)}{\sum_{d=0}^{M_i-1}\Phi_i^{RF}(dK)} & s = nK \\ 0 & \text{Otherwise} \end{cases}, \tag{8.8}$$

where $\Phi_i^{RF}(s)$ is the marginal probability distribution of s for an arriving packet (*i.e.*, the probability that an arriving packet finds the buffer of node v_i in the states of the form $(s,.)$). Φ_i can be computed as follows

$$\Phi_i(s) = \sum_{t=0}^{\min\{K-1,s\}} P_i^{RF}(s,t). \tag{8.9}$$

Also, $\mathbf{S}_i(s,t)$ can be derived using the following relation for $n \in \{0, 1, \ldots, M_i - 1\}$

$$\mathbf{S}_i(s,t) = \begin{cases} \mathbb{G}(\acute{\varepsilon}_i) & s = 0 \\ \otimes^{\{(n-1)K+(K+1-t)\}}\mathbb{G}(\acute{\varepsilon}_i) & s = nK \\ 0 & \text{Otherwise} \end{cases}.$$

Finally, $\mathbf{F}$ can be calculated by taking the average over all the conditional delay distributions $\mathbf{L}(s,t)$ as

$$\mathbf{F} = \pi_{h-1}(0,0)\mathbf{L}(0,0) + \sum_{d=1}^{m_{h-1}-1}\sum_{t=0}^{K-1}\pi_{h-1}(dK,t)\mathbf{L}(dK,t),$$

where $\mathbf{L}(s,t)$ is the probability distribution of the time taken for the whole block to be conveyed to the destination node given that the first packet of that block found the buffer of node v_{h-1} in state (s,t) when arrived. Note that after receiving the first packet of a block, node v_{h-1} has to wait until all its previously stored blocks are conveyed to the destination, during which some of the packets of the corresponding block might have already been arrived. Let $\mathbf{V}(x,y)$ be the probability distribution of the time to convey y innovative packets to the next node when x of those packets $(x \leq y)$ has yet to arrive for the same block, knowing that a packet departs with probability P_{out} and an innovative packet arrives with probability P_{in} in each time epoch. Hence, $\mathbf{L}(s,t)$ is derived using the following relation for $n \in \{0, 1, \ldots, M_i - 1\}$

$$\mathbf{L}(s,t) = \begin{cases} \mathbf{V}(K-1,K) & s = 0 \\ \{\otimes^{\alpha(n,t)}\mathbb{G}(\varepsilon_h)\} + \mathbf{V}(K-\beta,K) & s = nK \\ 0 & \text{otherwise} \end{cases},$$

where $\alpha(n,t) = (n-1)K + (K-t)$ is the number of packets that have to leave node v_{h-1} before the next block to be served and $\beta = \min\{K-1, \left\lfloor \frac{\alpha(n,t)P_{in}}{P_{out}} \right\rfloor\}$ is the expected number of packets from the corresponding block that arrived during the time when those $\alpha(n,t)$ packets were being conveyed. Further, $P_{in} = \overline{\varepsilon}_{h-1}(1 - \sum_{t=0}^{K-1} P^{TF}_{h-1}(t,t))$ for $h > 2$, $P_{in} = \overline{\varepsilon}_{h-1}$ for $h = 2$, and $P_{out} = \overline{\varepsilon}_h$. $\mathbf{V}(x,y)$ will be determined by the following lemma.

Lemma 8.1 $\mathbf{V}(x, y)$ *(defined for $x \leq y$) is the solution to the following equation,*

$$\mathbf{V}(x, y) = [\frac{p_1\mathbf{V}(x, y-1) + p_2\mathbf{V}(x-1, y) + p_3\mathbf{V}(x-1, y-1)}{p_1 + p_2 + p_3}] \otimes \mathbb{G}(p_4)$$

with boundary conditions:

$$\begin{array}{rcl} \mathbf{V}(0, y) & = & \otimes^y \mathbb{G}(1 - P_{out}) \\ \mathbf{V}(x, x) & = & \mathbb{G}(1 - P_{in}) \otimes \mathbf{V}(x-1, x) \end{array}, \tag{8.10}$$

where,

$$\begin{array}{ll} p_1 = P_{out}(1 - P_{in}) & p_2 = P_{in}(1 - P_{out}) \\ p_3 = P_{in}P_{out} & p_4 = (1 - P_{in})(1 - P_{out}). \end{array} \tag{8.11}$$

8.5 Results of Simulation

In this section, we compare our analytical results to the actual simulations. To study the effect of buffer size on throughput and block delay, we simulated a line network of eight hops for two cases where all the links have the same probability of erasure of 0.1 or 0.2. The buffer size m (in packets) is divided into M blocks of K packets.

Fig. 8.2 presents the variation of our analytical results and the actual simulations for both throughput and average delay of a block, as the buffer size m of the intermediate nodes is varied while the block size is fixed at $K = 5$ packets. It can be seen that as the buffer size is increased, average delay also increases linearly. It appears that above memory sizes of 10, the gain in capacity is negligible, while the latency increases significantly. Hence, there is no need to allocate more storage to the flow even if the space is available in the router. Further, for buffer sizes of less than 10 packets, there is a gap from the min-cut capacity, a diverging point from asymptotic results due to the finite buffer effect.

Fig. 8.3 presents a comparison between the actual and the estimated delay profile for a five-hop line network with the erasure probability on every link set to 0.05. It also compares the delay profiles for different buffer sizes when $K = 4$. It is noticed that as the buffer size of the intermediate nodes is increased, both average delay and its standard deviation are increased. This is undesirable since any increase in the standard deviation of the delay can make congestion control algorithms unstable.

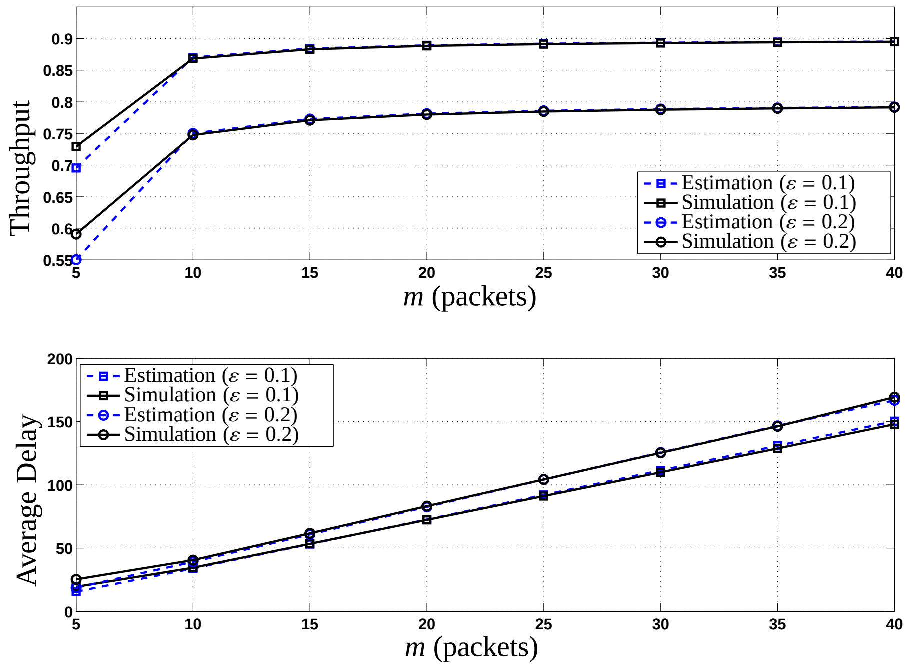

Figure 8.2: Delay and throughput of an 8-hop line network as a function of m.

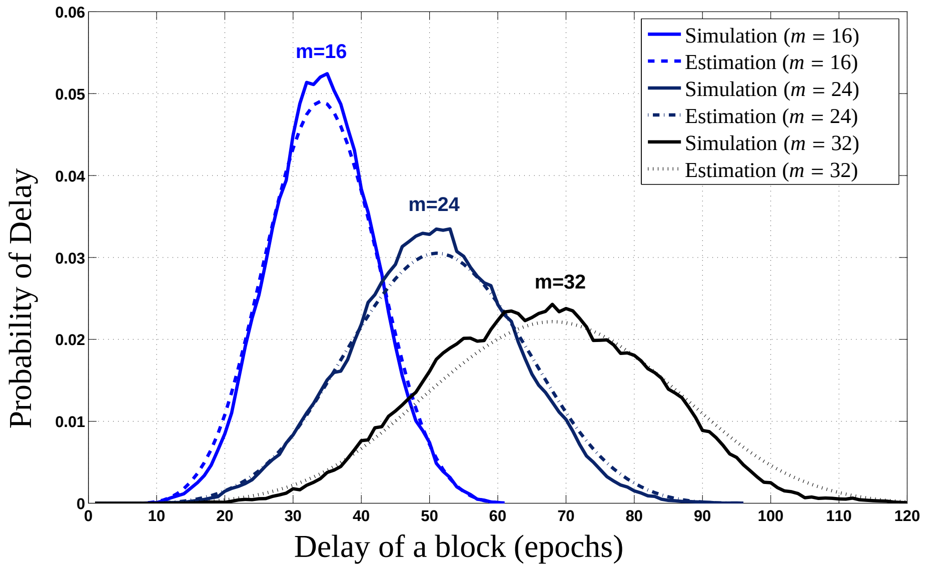

Figure 8.3: Delay profiles of a 5-hop line network for varying buffer sizes ($K = 4$).

CHAPTER 9

EXACT MODELING OF THE PERFORMANCE OF FINITE-BUFFER RANDOM LINEAR NETWORK CODING

In this chapter, we present an exact model for the analysis of the performance of random linear network coding (RLNC) in general wired networks with finite buffers.[1] We assert that because of RLNC, the content of buffers have dependencies which cannot be captured directly using the classical results of queueing theory. Here, we model the performance of the network using Markov chains by a careful derivation of the buffer occupancy states and their transition rules.

9.1 Introduction and Motivation

It is well-known that linear network codes achieve the min-cut capacity of networks for unicast applications [52]. In fact, random linear codes over large Galois fields suffice to achieve the min-cut capacity [96, 97]. *Random linear network coding* (RLNC) has been shown to improve the performance in distributed settings with time-varying network parameters. In these networks, a distributed and packetized network coding scheme, where each node stores the received packets and forwards random linear combinations of the stored packets when required, was introduced in [98, 99]. As a result, for a network of nodes with no buffer limitations, all arriving packets at a node are stored and then used to generate new packets to send. Hence, there is no information loss. However, in this case, upon reception of a packet, a node has to determine whether the incoming packet is in the linear span of its previously stored packets or not. Further, for generating every coded packet, all stored packets need to be accessed. It is therefore desirable to have limited buffer sizes, since it limits the complexity of storage and coded packet generation process.

Our objective is to study the relation between throughput of RLNC and the buffer sizes

[1]This work is done in collaboration with my former lab-mates, Dr. Badri N. Vellambi and Ahmad Beirami [95].

of intermediate nodes in the small buffer regime. The first and the key step in our approach is to derive, using algebraic tools, the state of the buffers using which the dynamics of the network can be completely characterized. We then derive the state update rules for each transmission in the network. Finally, using the developed state space and update rules, we obtain the throughput of the network using Monte Carlo simulations and compare the results to the actual packetized implementation of RLNC. We believe the proposed modeling framework is a significant step towards developing a theoretical framework for computing the throughput capacity and the packet delay distribution in general finite-buffer wired networks.

9.2 Problem Setup and Challenges

We model the network by an acyclic directed graph $\overrightarrow{G}(V, \overrightarrow{E})$, where packets can be transmitted over a link $\overrightarrow{e} = (u, v)$ only from the node u to v. The system is analyzed using a discrete-time model; each node can transmit at most one packet over a link in an epoch. The loss process on each link is assumed to be memoryless, i.e., packets transmitted on a link $\overrightarrow{e} = (u, v) \in \overrightarrow{E}$ are lost randomly with a probability of $\varepsilon_{\overrightarrow{e}} = \varepsilon_{(u,v)}$. Each node $v \in V$ has a buffer size of m_v packets with each packet having a fixed size. Source and destination are assumed to be able to store an infinitude of packets. Node s and node d represent the source and the destination nodes, respectively. The unicast information-theoretic throughput capacity is also defined as the expected rate (in packets/epoch) at which information packets are transferred from the source to the destination when the network is in steady-state. In other words, if τ_k is the time it takes for k information packets to be transmitted to the destination, the throughput capacity is given by

$$C(\overrightarrow{G}) = \lim_{k \to \infty} \frac{k}{\tau_k}. \tag{9.1}$$

There are two key challenges in finite-buffer networks. The first challenge is the choice of optimal buffer management strategy, which also depends on the routing/coding scheme that is in use. Due to losses on links, and finiteness of buffers, transmission of a packet by a node u on $\overrightarrow{e} = (u, v)$ does not guarantee successful reception by the node v. Thus, in the absence of any feedback, a node u does not know if it can delete a packet from its buffer to make room for its next incoming packet. Further, it is also unclear if transmitting a packet via several parallel paths will increase the throughput of the system. The second challenge in these networks is the following. Due to possible replication of packets in the network, it is neither possible to model the system dynamics by a simple queueing model where packets are customers and the buffers as queue sizes, nor is it feasible to treat the packets as flows in the network.

Random Linear Network Coding (RLNC) attractively bypasses these two challenges. It eliminates the need for a feedback strategy to delete the stored packets because the physical act of storing a packet becomes immaterial. It also eliminates the need for active replication by allowing transmitted/stored packets to be treated as elements of an abstract vector space. This makes RLNC a favorable choice for practical schemes in finite-buffer scenarios.

We consider the following packet-coding scheme introduced in [28], which is a finite-buffer adaptation of RLNC. In this scheme, at each epoch, random linear coding is used for both the packet generation and storage by intermediate nodes. As an example, consider a node u of buffer size m_u. At a given epoch, u generates an encoded packet by performing a random linear combinations of m_u stored data packets (over a sufficiently large Galois field[2] $\mathbb{F}_q$), and transmits the coded packet on an outgoing link. For storage, suppose a packet successfully arrives at v. Then, instead of storing the packet as is, node v multiplies the received packet by a random vector chosen uniformly from $\mathbb{F}_q^{m_v}$, and adds the resultant vector components to each of the present buffer contents.

Therefore, using RLNC, after just a single packet reception, the entire buffer becomes physically full with multiples of the received packet. Thus, even though the buffer of the node u is almost always physically full, the number of stored packets that is innovative with respect to any other subset of nodes can vary from 0 to m_u. As an example, when

[2]The size of the Galois field needs to be sufficiently large, in order to increase the chance of innovativeness of the coded packet.

performing RLNC, suppose that two nodes *a* and *b* receive and store two packets each generated from three original information packets from a relay *c*. In this case, *a* and *b* will have two innovative packets each for the destination. Now, suppose *a* delivers a packet to the destination. Then, *b* still contains two innovative packets for the destination. However, if *a* delivers another packet to the destination, *b* will only have one innovative packet for the destination, since both nodes together originally possessed only three innovative packets for the destination. In this example, the challenges of tracking the number of innovative packets and the interdependency between buffer contents gets compounded further as the packets from *a* and *b* are propagated to the other intermediate nodes. This interdependency between buffer contents signals the need for a novel notion of *occupancy* to track the number of innovative packets each node has for the destination, and hence to determine the throughput capacity of the network. This notion will be formalized in the following section.

The main motivating factor to develop a theoretical model for these networks is to understand the throughput capacity under RLNC. In order to measure the throughput of RLNC in these networks, one option is to perform a Monte Carlo simulation where encoded packets are generated using coefficients in a large finite field $\mathbb{F}_q$, and buffer updates are performed upon each successful reception. This is a significantly time-consuming simulation due to large field operations. A theoretical model tracking buffer dynamics based on occupancy of buffers will be a simpler alternate means. As we will see, the developed model provides a more efficient way of measuring the performance of finite-buffer networks. Additionally, it provide us with intuitive insights on the dynamics of buffer updates, which is a major step towards computing performance metrics for such networks, and analyzing the key trade-offs among them.

9.3 Exact Modeling of Finite-buffer RLNC

Here, we introduce the tools and steps that enable us to track changes in the buffer contents of nodes.

To identify the throughput as defined in (9.1), we assume that the source possesses a sufficiently large block of packets that has to be transmitted to the destination. The first aim is to formalize the notion of buffer occupancy by investigating the dimension of span of the stored packets in the buffers. Let $\{T_1, T_2, \ldots, T_k\}$ be the original information packets at the source. Let $[n] \triangleq \{1, 2, \ldots, n\}$ denote the set of all intermediate nodes, where $n = |V| - 2$. Let $P_{i,j}(t)$ be the packet contained in buffer slot j of relay i at time epoch t, where $P_{i,j}(t) = \sum_{l=1}^{k} a_{i,j,l} T_l$, $i \in [n]$, $j \in [m_i]$, and $a_{i,j,l}$ is a coefficient in the chosen Galois field $\mathbb{F}_q$. Let $\mathcal{V}(S)(t) \triangleq \text{span}\{P_{i,j}(t) | \ j \in [m_i], i \in S\}$ for all $S \subseteq [n]$. To simplify the notations, we will drop the reference to time in $\mathcal{V}(S)(t)$ by using $\mathcal{V}(S)$. Also, we define $S^c \triangleq [n] \setminus S$.

Definition 9.1 *For any two subsets of the intermediate nodes $S, S' \subseteq [n]$, we define the* innovativeness *of S w.r.t. S' at time instant t as:*

$$I_{S \to S'} = \dim(\mathcal{V}(S)) - \dim(\mathcal{V}(S) \cap \mathcal{V}(S')). \tag{9.2}$$

In other words, $I_{S \to S'}$ gives the number of innovative packets that buffer contents of nodes in S can generate which cannot be generated by the contents of the buffers of nodes in S'.

Definition 9.2 *The occupancy vector $\{b_S\}_{S \subseteq [n]}$ of the network is defined*[3] *to be*

$$b_S \triangleq \dim(\mathcal{V}(S)) - \dim(\mathcal{V}(S) \cap \mathcal{V}(S^c)), \ \ S \subseteq [n]. \tag{9.3}$$

The following lemma shows that the knowledge of occupancy vector $\{b_S\}_{S \subseteq [n]}$ is equivalent to knowing the innovativeness of any subset of the relay nodes w.r.t. any other subset. This result significantly reduces the number of state space variables.

[3]The precise definition of the occupancy vector must consider the packets that have already reached $\{d\}$ by using $b_S \triangleq \dim(\mathcal{V}(S)) - \dim(\mathcal{V}(S) \cap \mathcal{V}(S^c \cup \{d\}))$. However, the inclusion of $\{d\}$ affects update rules only when dealing with the destination. For simplicity, the equivalent definition without the inclusion of $\{d\}$ is used in all cases not involving the destination.

Lemma 9.1 *For $S, S' \subseteq [n]$, $I_{S \to S'} = b_{S'^c} - b_{\{S \cup S'\}^c}$.*

The proof can be found in Section B.1 of Appendix B.

Since the occupancy vector provides the innovativeness of the contents of each node w.r.t the remaining nodes, we need to be able to track the dynamics of the occupancy vector for successful transmissions on links to complete the system modeling. To do so, let superscripts $-$ and $+$ denote the status of a system parameter before and after a successful packet transmission on a link. The following results derive the rules for updating the occupancy vector when successful transmissions occur. Throughout these results, we denote *whp*/*wlp* to qualify an event if its probability of occurrence can be made arbitrarily close to unity/zero by increasing the field size alone.

Lemma 9.2 *(*Source-to-Relay update*) The update rules when a relay i successfully receives a packet from s are as follows* whp*.*

- *If $i \in S \subseteq [n]$ and $b_{\{i\}} < m_i$, then $b_S^+ = b_S^- + 1$.*
- *If $i \notin S \subseteq [n]$, $b_{\{i\}} < m_i$ and $I_{\{i\} \to S^c \setminus \{i\}} = m_i$, then $b_S^+ = b_S^- + 1$.*
- *Otherwise, $b_S^+ = b_S^-$.*

The proof can be found in Section B.2 of Appendix B.

Lemma 9.3 *(*Relay-to-Relay update*) The update rules when relay j successfully receives a packet from relay i are as follows* whp*.*

- *If $i \in S \subseteq [n]$, $j \in S^c$, $I_{\{j\} \to S^c \setminus \{j\}} < m_j$ and $I_{\{i\} \to S^c} > 0$, then $b_S^+ = b_S^- - 1$.*
- *Otherwise, $b_S^+ = b_S^-$.*

The proof can be found in Section B.3 of Appendix B.

Lemma 9.4 *(*Relay-to-Destination update*) The update rules when d successfully receives a packet from relay j are as follows* whp*.*

- *If $i \in S \subseteq [n]$ and $I_{\{i\}\to S^c} > 0$, then $b_S^+ = b_S^- - 1$.*
- *Otherwise, $b_S^+ = b_S^-$.*

The proof can be found in Section B.4 of Appendix B.

On the whole, an update of buffer occupancy occurs only when the delivered packet is innovative for the receiving node and the buffer of the receiving node is not full. Next, we describe how the state update rules could be utilized to obtain the throughput of a network. Let $\overrightarrow{E}^* = (\overrightarrow{e}_1, \ldots, \overrightarrow{e}_{|\overrightarrow{E}|})$ be an ordering of the edge set $\overrightarrow{E}$, and let $l(t) \in \{0,1\}^{|\overrightarrow{E}|}$ represent the realization of the channels at time t. That is $l_i(t) = 1$ if the i^{th} edge $\overrightarrow{e}_i$ in $\overrightarrow{E}^*$ does not erase the transmitted packet during the epoch t. Then, given the occupancy vector $\{b_S(t)\}_{S\subseteq[n]}$ and the channel realization $l(t)$, the occupancy vector $\{b_S(t+1)\}_{S\subseteq[n]}$ can be determined using the state update rules presented in Lemmas 9.2, 9.3, 9.4.

Further, the state transition probability matrix $\mathbb{T}$ for the corresponding Markov chain can be identified as follows. Also, let $T_{\overrightarrow{e}}$ be the state transition matrix given a successful packet transmission on the link $\overrightarrow{e}$. For any $\overrightarrow{e} \in \overrightarrow{E}$, $T_{\overrightarrow{e}}$ can be determined using Lemmas 9.2, 9.3, 9.4. Therefore,

$$\mathbb{T} = \sum_{l\in\{0,1\}^{|\overrightarrow{E}|}} \Big(\prod_{j:l_j=0} \varepsilon_{\overrightarrow{e}_j}\Big)\Big(\prod_{i:l_i=1} \bar{\varepsilon}_{\overrightarrow{e}_i} T_{\overrightarrow{e}_i}\Big). \tag{9.4}$$

This Markov chain can be proved to be *irreducible, aperiodic*, and *ergodic* [100][4]. Therefore, it possesses a unique steady-state probability distribution. Moreover, due to ergodicity, the time averages are equivalent to the statistical averages. Therefore, the throughput capacity $C(\overrightarrow{G})$ can be determined using the steady state probability of the event that the network is in a state wherein the nodes possessing a link to the destination have innovative packets as follows.

[4]The proof for the case of a line network is presented in [31].

$$C(\overrightarrow{G}) = \sum_{l\in\{0,1\}^{|\overrightarrow{E}|},\{b_S(t)\}} \mathfrak{R}(l, \{b_S(t)\}) \cdot \Pr\left(\{b_S(t)\}\right), \tag{9.5}$$

where $\mathfrak{R}(l, \{b_S(t)\})$ represents the number of successfully transmitted packets when state $\{b_S(t)\}$ and channel realization l occur together.

9.4 State Size Reduction in a Class of Networks

In Section 9.3, we observed that the number of state variables that we need to track at each time epoch is $2^n - 1$ since b_S, the innovativeness of every subset of relay nodes w.r.t. its complement, must be considered. In this section, we define a class of acyclic network for which the number of state variables is significantly smaller than $2^n - 1$ and hence the complexity of the modeling is considerably reduced.

Consider a partition of the set of relay nodes into types $\{H_1, H_2, \ldots\}$, where a relay node v belongs to H_k if the shortest hop-distance from v to the destination d is k, and $H_0 \triangleq \{d\}$. Define a class of networks $\mathcal{N}$ as those where there exists no link (v, v') such that $v \in H_k, v' \in H_{k'}$ and $k < k'$. Figure 9.1 illustrates a network from this class. Intuitively, in such a network, a link can exist only if it is between nodes of the same type of the partition, or start in a node belonging to a type with a higher index and end in a node belonging to a type with a lower index. This structure enables us to track significantly lesser number of innovativeness components as stated in the following result.

Theorem 9.1 *If the network belongs to $\mathcal{N}$, then we need only track $I_{S\to S'}$ where (1) $S \subseteq H_k$, and (2) $S' \subseteq \cup_{0\le k'\le k} H_{k'}$ such that $\cup_{0\le k'<k+1} H_{k'} \subseteq S'$ and $S \nsubseteq S'$.*

Therefore, in these networks, we only need to track $\left(\sum_{k>1}[2^{2|H_k|+|H_{k-1}|} - 3^{|H_k|}2^{|H_{k-1}|} - 2^{|H_k|} + 1]\right)$ state variables. As an example, line networks belong to $\mathcal{N}$. Hence, in a line network with n intermediate nodes, the number of state variables reduces to n, where they are exactly the same state variables developed in [31].

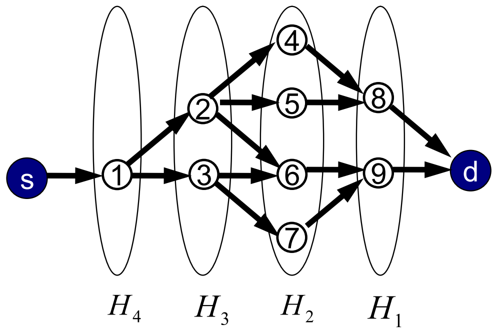

Figure 9.1: An example of a directed acyclic network in $\mathcal{N}$.

9.5 Simulation Results

In this section, we present the results of our performance modeling framework using state update rules in comparison with an actual packetized implementation of RLNC, and will show that our framework accurately models the buffer dynamics of the network.

We consider Network 1 and Network 2 shown in Figure 9.2 to compare the results of our simulations. In Network 1, the edges have erasure probabilities $\varepsilon_{(s,1)} = 0.1$, $\varepsilon_{(1,2)} = 0.6$,

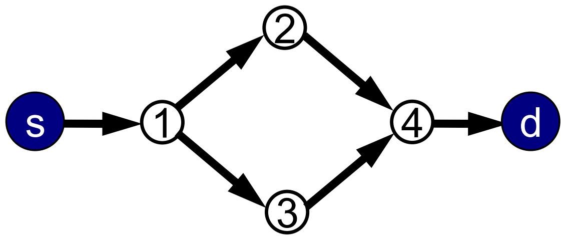

Figure 9.2: Network 1.

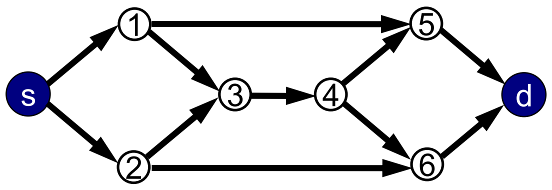

Figure 9.3: Network 2.

$\varepsilon_{(1,3)} = 0.5$, $\varepsilon_{(2,4)} = 0.4$, $\varepsilon_{(3,4)} = 0.5$, and $\varepsilon_{(4,d)} = 0.1$. In Network 2, all the edges have

$\varepsilon = 0.5$ except the edges $\{(s,1),(s,2),(5,d),(6,d)\}$ for which $\varepsilon = 0.25$. All the intermediate nodes are assumed to have the same buffer size. In order to measure the exact performance parameters of this network, a block of size $k = 10^5$ packets is sent from the source to the destination. Figure 9.4 and Figure 9.5 present the variations of the throughput measured

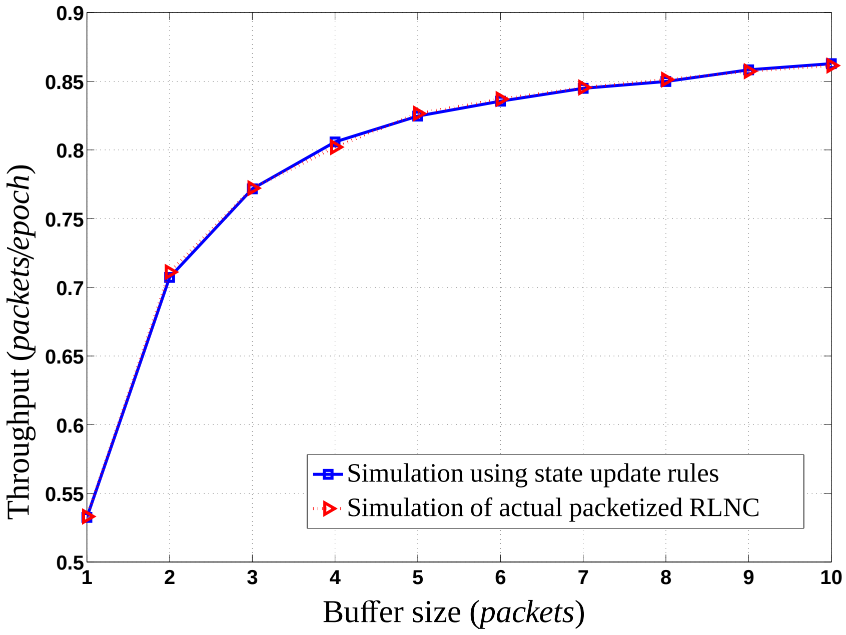

Figure 9.4: Throughput of Network 1 for different buffer sizes.

by actual simulation of RLNC and the throughput measured by simulation based on the state update rules developed in our work versus the buffer size. As it can be observed, our model is very close to the actual simulation results. Further, it confirms the optimality of RLNC for the infinite buffer setting as the curve approaches to the min-cut capacity for both networks. It is notable that the time it takes for the actual simulation of RLNC to be completed is roughly 1000 times the time it takes to simulate the state update rules.

Another important observation is presented in Table 9.1 which compares the number of states actually visited (identified by simulations), and a crude upper bound on the number of states in the Markov chain model. For Network 1, the number of state variables is $2^4 - 1 = 15$, and a provable upper bound for the number of states is $(m+1)^{15}$, where m is the buffer size of each intermediate node. However, it is noticed from simulations that the

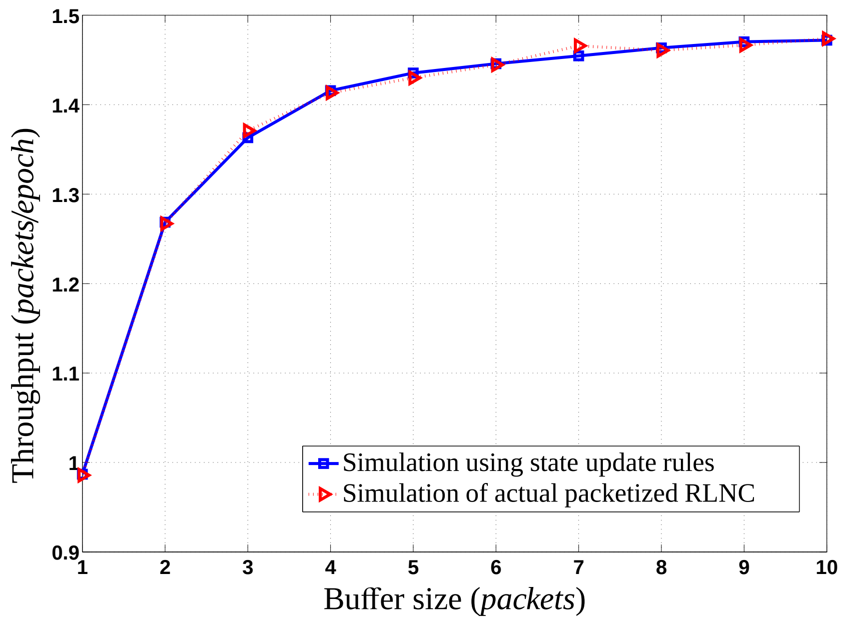

Figure 9.5: Throughput of Network 2 for different buffer sizes.

Table 9.1: Variation of the number of active states vs. buffer size in Network 1.

Buffer Size	No. of Active States	Upper Bound $(m+1)^15$
1	44	32768
2	600	14348907
3	4358	1073741824
4	21061	30517578125

number of states that is actually realized is much lesser than the bound. This observation signals the need to have a closer look at the Markov chain to reduce its size.

CHAPTER 10

DECODABILITY ANALYSIS OF RANDOM LINEAR NETWORK CODING IN LINE NETWORKS

In this section, we will address the problem of decodability when RLC is performed on a stream of arriving packets. The following questions arise when addressing such a problem. What does decodability of a stream of arriving packets at the source mean? Which parameters in the network rule the behaviour of decoding? How do we guarantee the decodability of a stream of arriving packets? First, we clearly define the problem of decodability of a stream of arriving packets, and discuss its importance with some examples. Then, we will find the limits on the mean arrival rate and will find expressions for the average length of a decoded block of packets.

10.1 Notations and Definitions

We consider a memoryless packet arrival process with mean rate λ for the source which is able to accommodate infinitely many packets until they are decoded at the destination. The block of packets that are decoded will then be deleted from the source buffer. We consider a line network of hop-length h, a graph with vertex set $V = \{s = v_0, v_1, v_2, ..., v_{h-1}, d = v_h\}$ and edge set $\overrightarrow{e} = \{\{v_i, v_{i+1}\} : i = 0, ..., h-1\}$ with erasure probability ε_i on link $\{v_{i-1}, v_i\}$ for $i = 1, ..., h$. It is assumed that random linear coding (RLC) over $\mathbb{F}_q$ is performed at the source as well as the intermediate nodes, where $\mathbb{F}_q$ is the Galois field of size q. [1] Moreover, we employ the following notations. For any $x \in [0, 1]$, $\overline{x} \triangleq 1 - x$. Node s and node d represent the source and destination nodes, respectively.

10.2 Maximum Decodable Throughput

To answer the questions asked at the beginning of the chapter, first we have to identify the rules and conditions under which a block of RLC encoded packets is decoded at the

[1] Throughout this chapter, we only consider the case where q is sufficiently large.

destination. As an example, similar to the model in [28], assume the source (encoder) has a finite memory of size m. Further, the destination (decoder) receives packets directly from the source, *i.e.*, there is no relay node. For now, we define the state of the network as the difference between the number of packets arrived at the source and the number of packets received by the destination, *i.e.*, transmitted to the destination and not lost. At the beginning of the first time epoch, the memory of the source is empty and we are in state 0. We remain in this state until the first packet p_1 arrives. Suppose the next packet transmitted from the source to the destination is not lost. Then we still remain in state 0, but the destination receives a packet that is a random linear combination of only the packet p_1, *i.e.*, a random scalar multiple of p_1. Hence, the decoder recovers p_1 from the received packet. Now suppose after the first packet p_1 arrives, the next outgoing packet is lost and we reach state 1. Suppose packet p_2 arrives before an outgoing packet is successfully transmitted, *i.e.*, transmitted and not lost. Then, any packet to be transmitted by the source is a random linear combination of p_1 and p_2. Suppose further that a packet is received by the destination, so we are again in state 1. This packet is currently useless to the destination node, since it is neither p_1 nor p_2. Nevertheless, it contains some information previously unknown to the destination node, *e.g.* p_1 and p_2 lie in a certain linear subspace. Consequently, the next packet received by the destination delivers previously unknown information, provided that it is linearly independent of the packet already stored. Such a packet is called an "innovative" packet. Further, it is notable that packets p_1 and p_2 will be decoded simultaneously at the destination and hence will generate a *decoded block* of length 2. Basically, every packet that is transmitted from a non-zero state is innovative at the destination because we assume q is sufficiently large. Also, every time the state returns to 0, a block of packets will be decoded and the length of the decoded block corresponds to the number of packets arrived during the time that the state was non-zero.

However, as claimed in [28], the statement above is true only when packets arrive at the source in states $0, 1, \ldots, m-1$. If a packet arrives at source in state m, the current contents of the source will be overwritten and hence corrupted, and will never be recovered. This is because a source with a buffer size m can only generate m innovative packets and after that any linear combination would be linearly dependent to the previously generated ones. In other words, the source has exactly m innovative packets to transmit before and after receiving the new packet, meaning that a packet worth of information is lost by this arrival. Moreover, the current contents of the source are corrupted and impossible to recover. In [28], the probability of packet loss is defined to characterize such behavior. In this work, however, an infinitely large buffer size is assumed for the source to investigate the characteristics and behavior of the decoding process at the destination for a multi-hop line network, without having to worry about packet loss or corruption of the contents of the source buffer. We will realize that such advantages come at the cost of decoding delay, *i.e.*, occasionally having to wait a long time for a block of packets to be decoded. We call a stream of packets with a fixed mean arrival rate λ *decodable* if the expected waiting time for a block of packets to be decoded is finite. The mean arrival rate λ associated with a decodable stream will be called a *decodable arrival rate*. However, the question is whether there are any arrival rates for which a stream of packets is not decodable. To answer this question, next, we will define parameters that have a critical role in characterizing decodability.

In the example above, we realize that each coded packet received at the destination is in fact a linear "equation" for which the original information packets arrived at the source are its "unknowns" to be found. Hence, upon receiving as many linearly independent equations at the destination as the number of unknowns, the system of linear equations is solvable and hence, a block of packets is decoded. The size of the decoded blocks is equal to the number of unknowns at the moment the system of linear equations is solved. Therefore, to guarantee that a stream of packets with arrival rate λ is decodable, the number of unknowns received at the destination should not grow unboundedly with respect to the number of equations received. To address such a problem, we need to be able to characterize the growth rate and dependencies of both the number of innovative packets at the destination (equations) and the number of original packets used in those innovative packets (unknowns). Previously, in Chapter 3, analytical results have been developed regarding the arrival rate of innovative packets at the destination when the network performs at steady state, *i.e.*, throughput.

In a line network setting, we define the innovativeness of node v_i with respect to node v_{i+1} at time epoch t, denoted by $I_i(t)$, as the number of packets stored in v_i that are innovative for v_{i+1}. The innovativeness of a node is limited to its buffer size, *i.e.*, $0 \leq I_i(t) \leq m_{v_i}$. Further, each arrival at the source increases its innovativeness, $I_S(t)$, by one. With RLC being performed on potentially a large number of information packets at the source, the buffer of the intermediate nodes contains a limited number of linearly independent packets (equations) including a large number of source-originated Packets (unknown variables). For the purpose of decoding analysis, in addition to the innovativeness of each node, the number of original packets involved in the buffer contents of each intermediate node is also considered. Hence, we define $P_i(t)$ as the number of original packets used in forming the linear combinations stored at the buffer of node v_i.

10.2.1 Decodability condition for a Two-hop Line Network

In this section, for simplicity of representation, we consider three nodes: A source S, a relay R, and a destination D. Further, their innovativeness are denoted by $I_S(t)$, $I_R(t)$, $I_D(t)$, and the number of original packets used in forming the linear combinations stored at their buffers are denoted by $P_S(t)$, $P_R(t)$, $P_D(t)$, respectively.

Previously, we have seen how the innovativeness of each node changes with arrival and/or departure of packets. For example, I_S increases by one with each packet arrival at the source, but potentially[2] decreases by one when a packet is transmitted successfully

[2] I_S is non-negative.

while $I_R < m$, where m is the buffer size of the relay R. Further, I_R potentially[3] increases by one if a packet is successfully transmitted to the relay from the source when $I_S > 0$, and potentially decreases by one when a packet is transmitted successfully to the destination. Finally, I_D only increases by one if a packet is successfully transmitted to the destination from the relay node when $I_R > 0$.

The changes in parameters $P_S(t)$, $P_R(t)$, $P_D(t)$ are quite different from how innovativeness of each node behaves. For the source node, since each arriving packet contributes a new unknown variable for decoding, $P_S(t)$ increases by one with each packet arrival at the source. Further, $P_R(t)$ either remains the same or takes the value of $P_S(t-1)$ where the latter occurs when a packet is received at relay R from the source no matter what are the buffer contents. In other words, when a packet is transmitted by the source and not lost, it brings a linear combination of all the packets stored at the source and combines it with the previously stored contents of the relay. Similarly, $P_D(t)$ either remains the same or takes the value of $P_R(t-1)$ where the latter occurs when a packet is received at the destination. Note that, the above changes occur regardless of the innovativeness of the packets. To summarize, let $B_p(t)$ be a Bernoulli random variable taking the value 1 with probability p at time epoch t and the value 0 otherwise. The following represents the changes in $P_S(t)$, $P_R(t)$, $P_D(t)$ in terms of λ, ε_1, and ε_2.

$$P_S(t+1) = P_S(t) + B_\lambda(t) \tag{10.1}$$

$$P_R(t+1) = P_R(t) + B_{\overline{\varepsilon}_1}(t)\,(P_S(t) - P_R(t)) \tag{10.2}$$

$$P_D(t+1) = P_D(t) + B_{\overline{\varepsilon}_2}(t)\,(P_R(t) - P_D(t)) \tag{10.3}$$

The following lemma summarizes the necessary and sufficient conditions for a block of packets to be decoded using the parameters described above.

Lemma 10.1 *A block of length K is decoded at time t^* if and only if both the following relations hold:*

[3]I_R should not exceed m.

1. $P_D(t^*) = I_D(t^*)$

2. $I_D(t^*) - I_D(t_0) = K$*, where* $t_0 = \max\left(\{t < t^* : P_D(t) = I_D(t)\}\right)$

The proof can be found in Appendix C.1.

Lemma 10.1 only presents the conditions for a single event of decoding of a block of packets. However, we are more interested in conditions that must hold to ensure the decodability of a stream of packets in the long run. At the beginning of this section, using a toy example, we observed that every time the state of the source returns to 0, a block of packets will be decoded. Although this statement is not true for a general multi-hop line network, later we will see that at steady-state, a block of packets is decoded if and only if the source revisits the state 0 at least once before the moment of decoding. Lemma 10.3 will present the necessary and sufficient condition for decodability at steady-state.

Lemma 10.2 *The ordered tuple* $(I_S(t), I_R(t))$ *forms an irreducible Markov chain.*

The proof can be found in Appendix C.2.

Lemma 10.3 *A stream of packets with source arrival rate* λ *is decodable if and only if in the Markov chain* $(I_S(t), I_R(t))$*, any state of the form* $(0, Y)$ *is recurrent, where* $Y = 0, 1, \ldots, m$*.*

The proof can be found in Appendix C.3.

Theorem 7.1 introduces a powerful tool to simplify and analyze complicated Markov chains with a large number of states [89]. We will use Theorem 7.1, to reduce the dimensions of the Markov chain defined in Lemma 10.2 as presented in the following corollaries.

Corollary 10.1 *The Markov chain* $(I_S(t), I_R(t))$ *can be collapsed into a new Markov chain* $I_R(t)$ *which represents the set of states of the form* $(X, I_R(t))$*, where* $X = 0, 1, \ldots$*.*

Corollary 10.2 *The Markov chain* $(I_S(t), I_R(t))$ *can be collapsed into a new Markov chain* $I_S(t)$ *which represents the set of states of the form* $(I_S(t), Y)$*, where* $Y = 0, 1, \ldots, m$*.*

Lemma 10.4 simplifies the condition of decodability introduced in Lemma 10.3 to include only the collapsed Markov chain $I_S(t)$ instead of the Markov chain $(I_S(t), I_R(t))$.

Lemma 10.4 *All the states of the Markov chain $(I_S(t), I_R(t))$ are recurrent if and only if all the states of the collapsed Markov chain $I_S(t)$ are recurrent.*

The proof can be found in Appendix C.4.

The following assumption is used to approximate the limit on the arrival rate λ. However, the assumption is not needed to prove the existence of such a limit. To avoid confusion and simplify the presentation, the Markov chains $I_S(t)$, and $I_R(t)$, are considered to be receive-first. The method of deriving both the transmit-first and receive-first distributions are described in details in Chapter 4.

Assumption 10.1 *Let $\Pr\{(I_S, I_R)\}$, $\Pr\{I_S\}$, $\Pr\{I_R\}$ be the steady-state probability distributions of the Markov chains $(I_S(t), I_R(t))$, $I_S(t)$, and $I_R(t)$, respectively. Then, the steady-state probability distributions of source and relay are independent of each other,* i.e., $\Pr\{(I_S, I_R)|I_S\} = \Pr\{I_R\}$, *and* $\Pr\{(I_S, I_R)|I_R\} = \Pr\{I_S\}$.

Finally, the following results summarizes the decodability condition in terms of the source arrival rate λ.

Lemma 10.5 *In the collapsed Markov chain $I_R(t)$, the steady state probability $\pi_R(m) = \lim_{t\to\infty} \Pr\{I_R(t) = m\}$ is a non-decreasing continuous function of λ, achieving its maximum, $\pi_R^{max}(m)$, when all the states in the collapsed Markov chain $I_S(t)$ are transient or null-recurrent.*

The proof can be found in Appendix C.5.

Theorem 10.1 *A stream of packets with source arrival rate λ is decodable if and only if $\lambda < C^*$, where C^* is the maximum throughput,* i.e., $C^* = \overline{\varepsilon}_1 \overline{\pi}_R^{max}(m)$.

The proof can be found in Appendix C.6.

10.3 Decoding Delay

In section 10.2, the existence of an upper limit to the decodable arrival rate λ is proved and derived. However, as mentioned before, decodability with no packet loss or corruption of the buffer contents, comes at the cost of decoding delay. In this section, we address the problem of finding analytical expressions for the average length of decoded blocks and its variations with arrival rate λ. The average length of a decoded block is a measure of decoding delay at the network since a larger decoded block implies a larger average packet delay.

First, we start with the familiar two-hop example and propose an upper bound on the average length of decoded blocks. Then, we will generalize the bound for a multi-hop line network.

10.3.1 Average Length of Decoded Blocks: Two-hop Line Network

Given a stream of packets is decodable, the Markov chain $I_S(t)$ is ergodic and therefore, has a steady-state probability distribution, denoted by $\pi_S(.)$, where $\pi_S(i) = \lim_{t\to\infty} \Pr\{I_S(t) = i\}$ for $i = 0, 1, 2, \ldots$. Further, the steady-state probability distribution for the Markov chain $I_R(t)$ is denoted by $\pi_R(.)$, where $\pi_R(i) = \lim_{t\to\infty} \Pr\{I_R(t) = i\}$ for $i = 0, 1, \ldots, m$. Finally, Theorem 10.2 provides an upper bound on the average length of a decoded block in a two-hop line network setting.

Lemma 10.6 *Let T_0^+ be the time to return to zero for the Markov chain $I_S(t)$,* i.e. $T_0^+ = \min\{t > t_0 : I_S(t) = I_S(t_0) = 0\}$. *Then, the expected time to return to zero at steady-state is* $E\left[T_0^+\right] = {\pi_S}^{-1}(0)$.

For the proof, See the proof of Lemma 5 in chapter 2 of [71].

Lemma 10.7 *Let $P_R^{dec}(k)$ be the probability that right after $I_S(t)$ returns to zero at time t_0,* i.e. $I_S(t_0) = 0$, *a block of packets including only the original packets arrived at the source* *up to time t_0 is decoded, given $I_R(t_0) = k$. Then, we have the following for $k = 1, 2, \ldots, m$:*

$$P_R^{dec}(k) > \{\varepsilon_1 \overline{\varepsilon}_2\}^{k-1}\ \overline{\varepsilon}_2\ e^{k\varepsilon_1\varepsilon_2}.$$

The proof can be found in Appendix C.7.

Theorem 10.2 *Let $\pi_R^{rcv}(k)$ be the conditional steady-state probability that $I_R(t) = k$ right after the relay receives a packet given that the relay is not full before the packet arrives,* i.e. *$I_R < m$. Let l_{dec} be the random variable representing the length of a decoded block. Then, the following provides an upper bound for the average length of a decoded block:*

$$E\left[l_{dec}\right] < \lambda E\left[T_0^+\right] \left\{ \sum_{k=1}^{m} \pi_R^{rcv}(k) P_R^{dec}(k) \right\}^{-1}. \qquad (10.4)$$

The proof can be found in Appendix C.8.

10.3.2 Average Length of Decoded Blocks: Multi-hop Line Network

Here, we extend the results of Section10.3.1 to a line network with h hops. The steady-state probability distribution for the Markov chain $I_j(t)$ corresponding to the relay v_j is denoted by $\pi_j(\cdot)$ for $j = 1, 2, \ldots, h$, where $\pi_j(i) = \lim_{t\to\infty} \Pr\{I_j(t) = i\}$.

Lemma 10.8 *Let $P_1^{dec}(k_1)$ be the probability that right after $I_S(t)$ returns to zero at time t_0,* i.e. *$I_S(t_0) = 0$, all the information required to decode the original packets arrived at the source up to time t_0 is passed to the relay node v_1, given $I_1(t_0) = k_1$. Similarly, Let $P_2^{dec}(k_2)$ be the probability that right after $I_1(t)$ becomes zero at a time t_1,* i.e. *$I_1(t_1) = 0$, all the required information to decode the original packets arrived at the source up to time t_0 is passed to the relay node v_2, given $I_2(t_1) = k_2$. Further, Let $P_3^{dec}(k_3), \cdots, P_{h-1}^{(k_{h-1})}$ be defined in a similar fashion, where h is the number of hops. Then, we have the following for $j = 1, 2, \ldots, h-1$ and $k_j = 1, 2, \ldots, m_i$:*

$$P_j^{dec}(k_j) > \left\{\varepsilon_j r_{j+1}\right\}^{k_j-1} r_{j+1}\, e^{k_j \varepsilon_j \overline{r}_{j+1}},$$

where, $r_j = \overline{\varepsilon}_j \overline{\pi}_j(m_j)$.

The proof can be found in Appendix C.9.

Theorem 10.3 *Let $\pi_j^{rcv}(k)$ be the conditional steady-state probability that $I_j(t) = k$ right after node v_j receives a packet given that the relay is not full before the packet arrives,* i.e. *$I_j < m$. Then, the following provides an upper bound for the average length of a decoded block in a line network of h hops:*

$$E\left[l_{dec}\right] < \lambda E\left[T_0^+\right] \prod_{j=1}^{h-1} \left\{ \sum_{k=1}^{m_j} \pi_j^{rcv}(k) P_j^{dec}(k) \right\}^{-1}. \tag{10.5}$$

The proof can be found in Appendix C.10

10.3.3 Simulation Results

In this section, the proposed upper bounds are validated by comparing it with simulations. In our simulation setup, the buffer size of all the relay nodes are assumed to be equal, $m = 5$ packets. Further, the probability of erasure on all the links are assumed to be the same, $\varepsilon = 0.1$. The mean arrival rate at the source, λ, is varied in a range that the stream of packets remain decodable. Then, the variations of the average length of a decoded block are presented in Fig. 10.1, Fig. 10.2, Fig. 10.3, and Fig. 10.4 for a line network with 2, 3, 4, and 5 hops, respectively. Clearly, from the simulation results, the upper bound is fairly tight for a two-hop line network, and as the number of hops increases, the upper bound becomes looser. The reason for such behavior is multiplication of the upper bound introduced for the two-hop case for a multi-hop scenario.

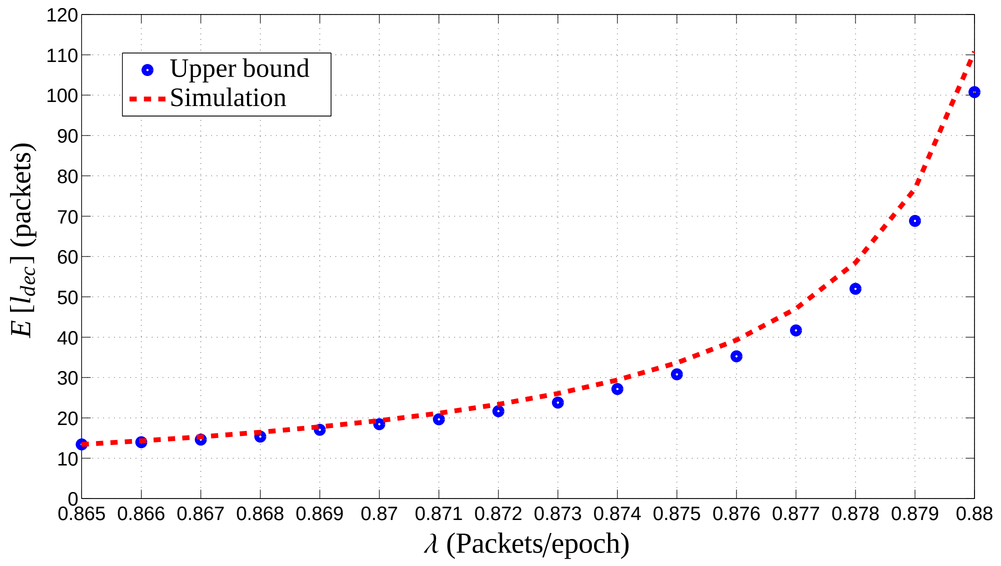

Figure 10.1: Variations of the average length of a decoded block in a two-hop line network

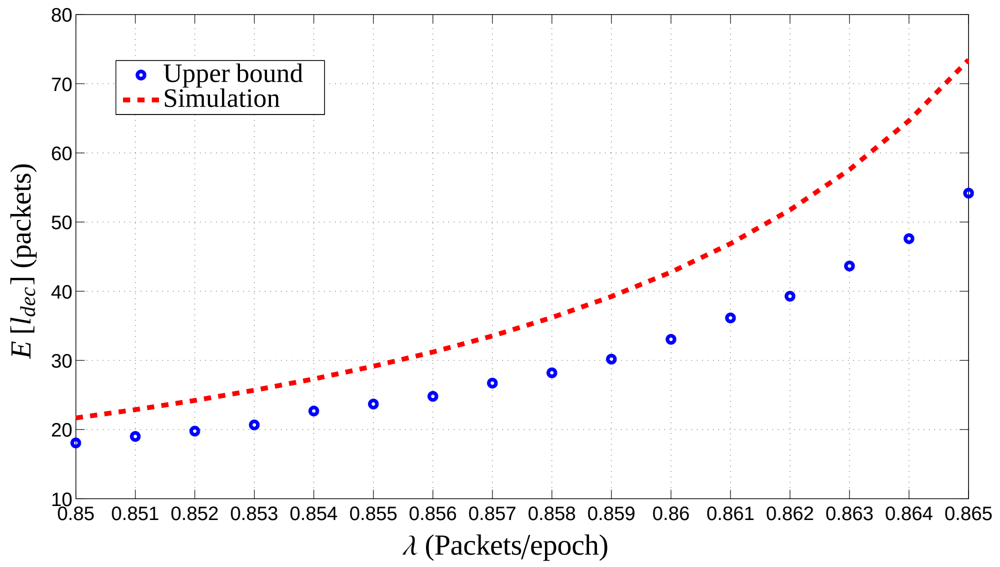

Figure 10.2: Variations of the average length of a decoded block in a three-hop line network

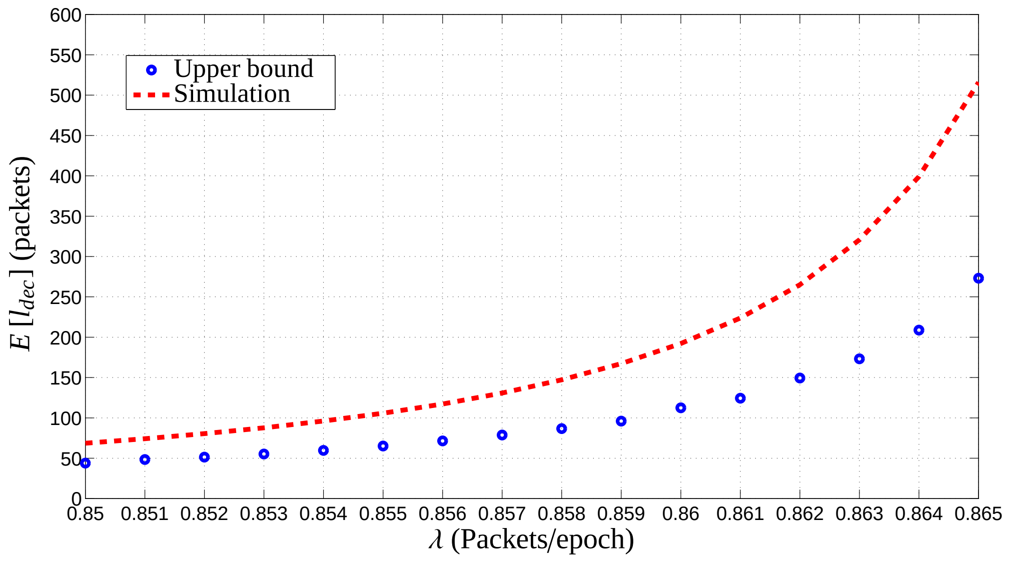

Figure 10.3: Variations of the average length of a decoded block in a four-hop line network

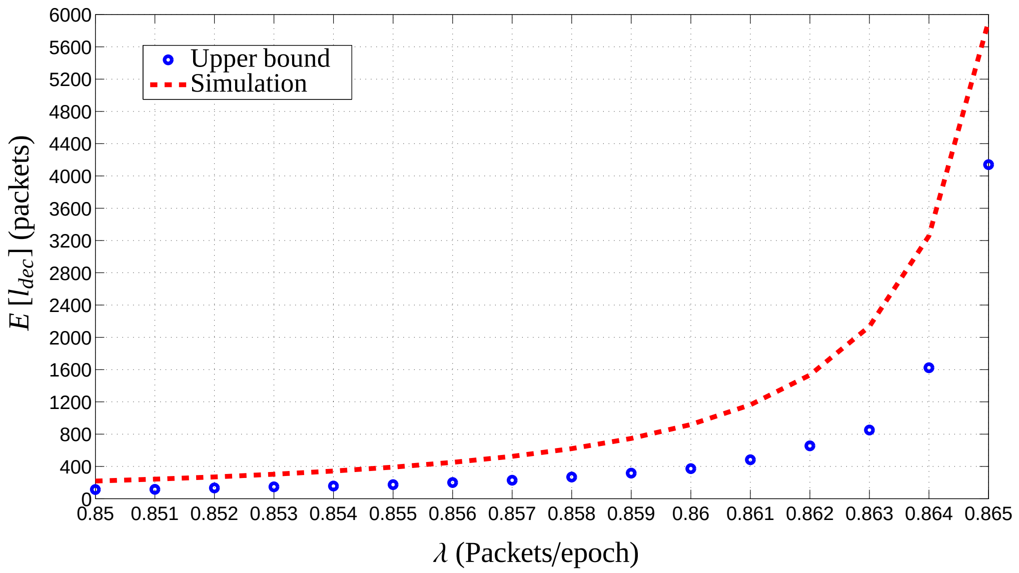

Figure 10.4: Variations of the average length of a decoded block in a five-hop line network

CHAPTER 11

CONCLUSION OF THE THESIS

In this dissertation, we investigate the problem of performance analysis in finite-buffer networks, where the throughput and packet delay are introduced as the main performance parameters to be characterized. The dissertation addresses issues ranging from the computation of network throughput and probability distribution of packet delay to modeling the network buffer dynamics when finite-memory random linear network coding is performed.

In Chapter 1, the problem of performance analysis in finite-buffer networks is motivated by presenting the related works in the literature. The performance measures are linked to the stationary distribution of an underlying irreducible Markov chain that exactly models the network dynamics. In Chapter 2, a general framework is proposed for studying the latency and throughput of different network scenarios (*e.g.*, wired/wireless, mobile/fixed topology) in finite-buffer regime. In particular, given the communication protocols and network settings, an iterative scheme is proposed to approximate the occupancy distribution of buffers by modeling the states of the network with Markov chains of appropriate size and complexity. These buffer occupancy distributions can then be used to determine packet delay and its interplay with network throughput. We believe that our developed framework can help to understand, and to design more suitable protocols for real-time applications with high speed (finite-buffer) routers. In Chapter 3, the proposed analytical framework is used to obtain analytical expressions for the throughput and probability distribution of packet delay in multi-hop line networks with erasure links. It is claimed that the problem of identifying capacity is directly related to the problem of finding schemes that are rate-optimal. Hence, the communication protocol is chosen to be hop by hop lossless feedback. However, rate-optimality can be achieved by employing random linear network coding over a large finite field in the absence of feedback. Using simulations, the proposed iterative techniques were noticed to be computationally-efficient and near-accurate to analyze and study the behavior of line networks. In Chapter 4, the performance parameters such as throughput and average latency were analyzed in general wired acyclic networks with erasure links when a random packet routing scheme with ideal feedback on the links is used. Here, the main difference with multi-hop line network setting is having nodes with multiple incoming and outgoing packet streams. In Chapter 5, the framework of analysis is tuned to include wireless erasure networks and investigate the trade-offs between throughput, average packet delay and buffer size when a modified backpressure routing policy is used. When dealing with backpressure routing scheme, the main difference in applying our iterative framework is to account for dependency of the arrival and departure rates on the current occupancy of each node. One of the main reasons that such a routing algorithm is chosen for analysis is its throughput-optimality for the infinite-buffer case. In Chapter 6, the effect of finite buffer size on the performance parameters of multihomed wireless networks is investigated along with the problem of buffer size optimization to meet the requirements of delay-sensitive applications. Here, the delay constraint is assumed such that at least a certain fraction of packets is required to reach the destination with a delay smaller than an application-dependent threshold value, and then, the delay-constrained throughput, also known as goodput, is considered for maximization. In Chapter 7, We have considered finite-buffer disruption-tolerant networks (DTNs) wherein a direct path between two particular nodes does not exist due to the mobility and sparseness of the nodes. Hence, the nodes will deliver messages from source to destination using a "store, carry, and forward" strategy. Our goal is to obtain analytical expressions for packet latency in such networks for any mobility model which has stationary properties. Since, the full state-space description of the network is very large, to reduce the state-space and simplify the analysis, we use the idea of chain-collapsing, meaning that, for a particular relay node and the source node, we identify all the "desirable" states which contribute to the delay problem, together with certain additional "auxiliary" states to arrive at an "embedded" Markov chain. Then, we developed two collapsed Markov chains RMC and SMC originated from the full state-space of the network. However, these two Markov chains are not only dependent on each other but also closely related. Further, their dependency leads us into solving both of them using our proposed iterative estimation technique. We have considered constraints posed by contention between nodes for wireless channel to obtain a more realistic model. Our analytical results are validated using simulations for mobility models such as two-dimensional random walk and the random waypoint mobility model. In Chapter 8, the problem of performance analysis for multi-hop line networks with erasures is extended to include a block-by-block random linear network coding scheme with feedback on the links which guarantees the length of each decoded block to be the same. In Chapter 9, a novel notion of buffer occupancy for finite-memory random linear network coding (RLNC) in wired networks is derived. Using this notion, a Markov-chain-based framework is developed to identify the throughput offered by RLNC using Monte Carlo simulations. This framework offers significant computational benefits over a complete simulation of RLNC. Though the size of the Markov chain is exponentially growing with the network size, simulations suggest that a very small portion of the state space is actually visited in reality. As future work, a closer look at the state space and a thorough analysis to reduce the state space can be performed to eventually derive analytical throughput estimates. In Chapter 10, the problem of decodability when RLNC is performed on a stream of arriving packets is addressed. We first define the decodability of a stream of arriving packets as the finiteness of the waiting time for a block of packets to be decoded at the destination. Further, we prove that for any mean source arrival rate smaller that the finite-buffer throughput capacity of a two-hop line network, the stream is decodable. finally, upper bounds are derived for the average decoded block length in multi-hop line networks, and validated by simulations.

APPENDIX A

PROOFS OF RESULTS IN CHAPTER 7

A.1 Proof of Lemma 7.4

Consider the following subsets of states in the original state-space description of the network.

- A: The source is in its on-period and its most recent event was a packet arrival.
- R: The source is in its off-period and its most recent event was meeting a non-full relay or the destination.
- R_F: The source is in its off-period and its most recent event was meeting a full relay.

Here, we collapse these subsets into just three states, resulting in the new Markov chain shown in Fig. A.1. Clearly, α_s from the original chain in Fig. 7.3 is given by the probability that the chain in Fig. A.1, starting from state A, visits state R before coming back to state A again. Similarly, β_s is given by the probability that the chain in Fig. A.1, starting from state R, visits state A before coming back to state R again. Such probabilities can be obtained from the fundamental matrix of the Ergodic Markov Chain (see chapter 2 of [71] for a discussion on the fundamental Matrix of an ergodic chain) in Fig. A.1. Let Z be the fundamental matrix for this chain. The probabilities α_s and β_s can be derived using

$$\alpha_s = \frac{\pi_R}{\pi_A \{Z_{RR} - Z_{AR}\} + \pi_R \{Z_{AA} - Z_{RA}\}},$$

$$\beta_s = \frac{\pi_A}{\pi_A \{Z_{RR} - Z_{AR}\} + \pi_R \{Z_{AA} - Z_{RA}\}}.$$

The results will follow after performing the necessary computation which would be computing the fundamental matrix Z for Markov chain shown in Fig. A.1.

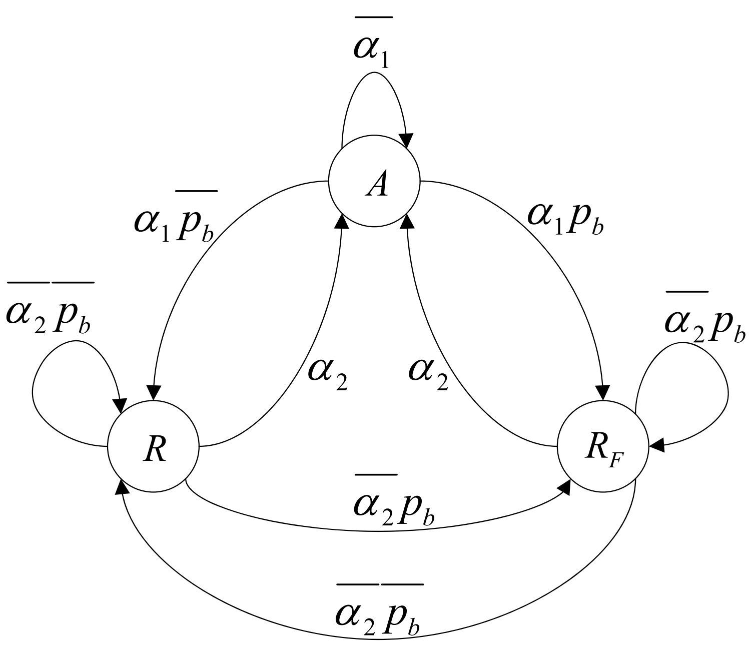
$\overline{\alpha_1}$
A
$\alpha_1 \overline{p_b}$
$\alpha_1 p_b$
$\overline{\alpha_2}\,\overline{p_b}$
$\overline{\alpha_2} p_b$
α_2
α_2
R
R_F
$\overline{\alpha_2} p_b$
$\overline{\alpha_2}\,\overline{p_b}$

Figure A.1: Three state MC for obtaining α_s and β_s

A.2 Proof of Lemma 7.6

Considering the network at steady-state, S_0 is the random variable representing the time until the next arrival at the source, given that the source is in its on-period at time $\tau = 0$. At this point, the random location of the other $n + 1$ nodes follows the steady-state spatial distribution of the mobility model. Hence, $T_{\infty,n}$ is the random variable representing the waiting time until the source comes in contact with one of the $n + 1$ nodes. Further, S_0 and $T_{\infty,n}$ are independent since the arrival process is independent of the mobility. Hence, the parameter α_1 can be expressed as

$$\begin{aligned}\alpha_1 &= \Pr\{T_{\infty,n} < S_0\} \\ &= \sum_{\tau=1}^{\infty} \Pr\{S_0 = \tau\} \Pr\{T_{\infty,n} < \tau | S_0 = \tau\} \\ &= \sum_{\tau=1}^{\infty} F_{T_{\infty,n}}(\tau) P_{S_0}(\tau).\end{aligned}$$

The results for the parameter α_2 can be proved in a similar fashion.

As for the parameter β_1, given that the source node has no contacts with any other node (relay or destination) at time $\tau = 0$, meeting a node during the next time epoch means that

$T_{\infty,n} = 1$ and hence the result follows. As for the parameter β_2, given that the source node is in contact with a node (relay or destination) at time $\tau = 0$, meeting with none of the other nodes during the next time epoch means that $T_{0,n} > 1$ and hence $\beta_2 = 1 - \Pr\{T_{0,n} = 1\}$.

A.3 Proof of Proposition 7.1

Let p_i be the probability that a packet is stored at the i^{th} buffer space of a relay node upon its reception from the source. In this case, such a packet needs to wait for $i - 1$ previously stored packets to be delivered to the destination before being served. Hence, for the packet to be delivered to the destination, that particular relay node must establish i links with the destination. Since upon receiving the packet the remaining $n + 1$ nodes follow the steady-state spatial distribution of the mobility model, the average waiting time to meet the destination for the first time is $E[T_\infty]$ epochs. Note that meeting the destination node is equivalent to establishing a link with it since there is no contention when the source and the destination are in the same communication range. Afterwards, the remaining $i - 1$ links will take an average time of $(i - 1)E[T_0]$ epochs to be established. Therefore, by taking an average, we have the following

$$D_{net} = P_z.(0) + \overline{P_z}.\left(\sum_{i=1}^{B} p_i\,(E[T_\infty] + (i-1)E[T_0])\right). \tag{A.1}$$

Further, p_i can be characterized as the conditional probability of having i packets in the buffer of a relay node given that the most recent link that the relay node had is with a non-empty source, and it can be obtained using

$$p_i = \frac{\Pr\{S_i\}}{\sum_{j=1}^{B} \Pr\{S_j\}}, \tag{A.2}$$

where, $\Pr\{S_i\}$ is known for $i = 1, 2, \ldots, B$ from the steady-state analysis of RMC. Next, P_z is the conditional probability of the source meeting the destination given that its most recent link was established with a non-full relay or the destination. After incorporating the contention between relays, P_z can be determined from

$$P_z = \frac{E[T_0]^{-1}}{\frac{n}{n+1}\overline{p_c}E[T_{0,n}]^{-1}}, \tag{A.3}$$

where, the contention failure probability p_c is derived in Lemma 7.3. Finally, by plugging (A.2) and (A.3) into (A.1) the result will follow.

A.4 Proof of Proposition 7.2

Let p'_i be the probability that a packet is stored at the i^{th} buffer space of the source node upon its arrival. In this case, such a packet needs to wait for $i-1$ previously stored packets to leave the source before being served. Hence, the source node must establish i links with a non-full relay or the destination. Since upon arrival of the packet the remaining $n+1$ nodes follow the steady-state spatial distribution of the mobility model, the average waiting time to meet a relay or the destination for the first time is $E[T_{\infty,n}]$ epochs. Further, because the relays might be full, the average waiting time to establish a link with a non-full relay or the destination for the first time would be $E[T_{\infty,n}]\overline{p_b}^{-1}$ epochs. Similarly, the remaining $i-1$ links will take an average time of $(i-1)E[T_{0,n}]\overline{p_b}^{-1}$ epochs to be established. Therefore, by taking an average, we have the following

$$D_{queue} = \overline{p_b}^{-1} \sum_{i=1}^{\infty} p'_i \left(E[T_{\infty,n}] + (i-1)E[T_{0,n}]\right), \tag{A.4}$$

Further, p'_i is the conditional probability of having i packets in the buffer of the source given that the most recent event at the source is a packet arrival, and it can be obtained from

$$p'_i = \frac{\Pr\{A_i\}}{\sum_{j=1}^{\infty} \Pr\{A_j\}}, \tag{A.5}$$

where, $\Pr\{A_i\}$ is known for $i = 1, 2, \ldots$ from the steady-state analysis of SMC. Finally, by plugging (A.5) into (A.4) the result will follow.

APPENDIX B

PROOFS OF RESULTS IN CHAPTER 9

B.1 Proof of Lemma 9.1

$$\begin{aligned}
b_{S'^c} - b_{\{S\cup S'\}^c} &= \dim(\mathcal{V}(S'^c)) - \dim(\mathcal{V}(S'^c)\cap\mathcal{V}(S')) \\
&\quad + \dim(\mathcal{V}(\{S\cup S'\}^c)\cap\mathcal{V}(S\cup S')) \\
&\quad - \dim(\mathcal{V}(\{S\cup S'\}^c)) && \text{(B.1)} \\
&= \dim(\mathcal{V}(S'^c\cup S')) - \dim(\mathcal{V}(S')) \\
&\quad - \dim(\mathcal{V}(\{S\cup S'\}^c\cup\mathcal{V}\{S\cup S'\})) \\
&\quad + \dim(\mathcal{V}(S\cup S')) && \text{(B.2)} \\
&= \dim(\mathcal{V}([n])) - \dim(\mathcal{V}(S')) \\
&\quad + \dim(\mathcal{V}(S\cup S')) - \dim(\mathcal{V}([n])) && \text{(B.3)} \\
&= \dim(\mathcal{V}(S\cup S')) - \dim(\mathcal{V}(S')) && \text{(B.4)} \\
&= \dim(\mathcal{V}(S)) - \dim(\mathcal{V}(S)\cap\mathcal{V}(S')) && \text{(B.5)} \\
&= I_{S\to S'} && \text{(B.6)}
\end{aligned}$$

Here, we used the fact that for any $A, B \subset [n]$, $\dim(\mathcal{V}(A)\cap\mathcal{V}(B)) = \dim(\mathcal{V}(A)) + \dim(\mathcal{V}(B)) - \dim(\mathcal{V}(A\cup B))$. ■

B.2 Proof of Lemma 9.2

First, we consider the case where $i \in S$. Let $\mathcal{A}^- = \{A_1^-, A_2^-, \ldots, A_{m_i}^-\}$, $\mathcal{B}^- = \{B_1^-, B_2^-, \ldots, B_{|\mathcal{B}^-|}^-\}$, $C^- = \{C_1^-, C_2^-, \ldots, C_{|C^-|}^-\}$ be the buffer contents of relay i, relays $S\setminus\{i\}$, and relays S^c, before update, respectively. Similar to the proof of Lemma 9.1,

$$b_S = \dim(\text{span}\{\mathcal{A}\cup\mathcal{B}\cup C\}) - \dim(\text{span}\{C\}). \tag{B.7}$$

Let E be the packet received by node i from the source, and is innovative for $\mathcal{A}$, $\mathcal{B}$, and C. We consider two cases:

- Case 1: Suppose there exist coefficients $\lambda_l, \theta_k, \pi_d$ s.t. $\lambda_l \neq 0$ for at least one l, and $\sum_l \lambda_l A_l^- + \sum_k \theta_k B_k^- + \sum_d \pi_d C_d^- = 0$. The existence of such coefficients is equivalent to $I_{\{i\} \to [n] \setminus \{i\}} < m_i$, which is equivalent to $b_{\{i\}} < m_i$. By similar arguments, we can show that $E \in \text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+ \cup C^+\}$ whp. Therefore, $\dim(\text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+ \cup C^+\}) = \text{span}\{\mathcal{A}^- \cup \mathcal{B}^- \cup C^-\} + 1$, which results in $b_S^+ = b_S^- + 1$.

- Case 2: Suppose no such $\lambda_l, \theta_k, \pi_d$ as in Case 1 exists. In this case, it can be shown that $\dim(\text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+ \cup C^+\}) \subseteq \dim(\text{span}\{\mathcal{A}^- \cup \mathcal{B}^- \cup C^-\})$, and hence $b_S^+ = b_S^-$.

Now, let $i \in S^c$. Here, let $\mathcal{A}^- = \{A_1^-, A_2^-, \ldots, A_{|\mathcal{A}^-|}^-\}$, $\mathcal{B}^- = \{B_1^-, B_2^-, \ldots, B_{m_i}^-\}$, $C^- = \{C_1^-, C_2^-, \ldots, C_{|C^-|}^-\}$ be the buffer contents of relays S, relay i, and relays S^c, before update, respectively. In this case, the only setting where a non-trivial change occurs can be shown to be the case when there exists no λ_l, θ_k s.t. $\lambda_l \neq 0$ for at least one l, and $\sum_l \lambda_l B_l^- + \sum_k \theta_k C_k^- = 0$, but there exists $\lambda_l, \theta_k, \pi_d$ s.t. $\lambda_l \neq 0$ for at least one l, and $\sum_l \lambda_l B_l^- + \sum_k \theta_k C_k^- + \sum_{k'} \pi_d A_{k'}^- = 0$. Here, it can be shown that $\dim(\text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+ \cup C^+\}) = \text{span}\{\mathcal{A}^- \cup \mathcal{B}^- \cup C^-\} + 1$ whp, and hence $b_S^+ = b_S^- + 1$ whp. In all other cases, the state vector remains unchanged whp. ■

B.3 Proof of Lemma 9.3

From Definition 9.2 it is clear that if $i, j \in S$, then $b_S^+ = b_S^-$. The same applies when $i, j \in S^c$. In the following, we investigate the update rule for the case $i \in S$, $j \in S^c$. For the case $i \in S^c$, $j \in S$, the update rule is $b_S^+ = b_S^-$ and the proof is similar to the one presented for the case $i \in S$, $j \in S^c$.

Hence, we only consider the case where $i \in S, j \in S^c$. Let $\mathcal{A}^- = \{A_1^-, A_2^-, \ldots, A_{m_i}^-\}$, $\mathcal{B}^- = \{B_1^-, B_2^-, \ldots, B_{|\mathcal{B}^-|}^-\}$, $C^- = \{C_1^-, C_2^-, \ldots, C_{m_j}^-\}$ and $\mathcal{D}^- = \{D_1^-, D_2^-, \ldots, D_{|\mathcal{D}^-|}^-\}$ be the buffer contents of relay i, relays $S \setminus \{i\}$, relay j, and relays $S^c \setminus \{j\}$ before packet transmission, respectively. Suppose packet $E = \sum_{l=1}^{m_i} \alpha_l A_l^-$ successfully transfers from relay i to relay

j. Then, for any $S \subseteq [n]$, We will have $\mathcal{A}^+ = \mathcal{A}^-$, $\mathcal{B}^+ = \mathcal{B}^-$, $\mathcal{D}^+ = \mathcal{D}^-$, and $C^+ = \{C_1^- + \beta_1 E, C_2^- + \beta_2 E, \ldots, C_{m_j}^- + \beta_{m_j} E\}$. Note that the coefficients α_l and β_k are chosen randomly from $\mathbb{F}_q$. Let $\mathcal{G}^- = \text{span}\{\mathcal{A}^-\} \cap \text{span}\{C^- \cup \mathcal{D}^-\}$. We consider two cases:

- Case 1: Suppose there exists λ_l, θ_k such that $\lambda_l \neq 0$ for at least one l and $\sum_l \lambda_l C_l^- + \sum_k \theta_k D_k^- = 0$. Hence,

$$\sum_l \lambda_l C_l^+ + \sum_k \theta_k D_k^- = (\sum_l \lambda_l \beta_l) E \in \qquad \text{span}\{C^+ \cup \mathcal{D}^+\}$$

  Therefore, $E \in \text{span}\{C^+ \cup \mathcal{D}^+\}$ whp. Further, if $\mathcal{G}^- \neq \text{span}\{\mathcal{A}^-\}$, then $E \notin \mathcal{G}^-$ whp, and $\text{span}\{C^+ \cup \mathcal{D}^+\} = \text{span}\{C^- \cup \mathcal{D}^- \cup \{E\}\}$. Hence,

$$\begin{aligned} b_S^+ &= \dim(\text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\}) \\ &\quad - \dim(\text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\} \cap \text{span}\{C^+ \cup \mathcal{D}^+\}) \\ &= \dim(\text{span}\{\mathcal{A}^- \cup \mathcal{B}^-\}) \\ &\quad - \dim(\text{span}\{\mathcal{A}^- \cup \mathcal{B}^-\} \cap \text{span}\{C^- \cup \mathcal{D}^- \cup \{E\}\}) \\ &= b_S^- - 1 \end{aligned}$$

  Note that $\mathcal{G}^- \neq \text{span}\{\mathcal{A}^-\} \Leftrightarrow I_{\{i\}\to S^c} > 0$, and the existence of such $\lambda_l, \theta_k \Leftrightarrow I_{\{j\}\to S^c\setminus\{j\}} < m_j$.

  On the other hand, if $\mathcal{G}^- = \text{span}\{\mathcal{A}^-\}$, then $E \in \mathcal{G}^-$ and since $\mathcal{G}^+ = \mathcal{G}^-$, we will have $\text{span}\{C^+ \cup \mathcal{D}^+\} = \text{span}\{C^- \cup \mathcal{D}^-\}$, and hence $b_S^+ = b_S^-$.

- Case 2: Suppose no such λ_l, θ_k as in Case 1 exist. Let $\mathcal{F}^- = \{F_i^-, i \in [|\mathcal{F}^-|]\}$ be a basis for $\text{span}\{\mathcal{A}^- \cup \mathcal{B}^-\} \cap \text{span}\{C^- \cup \mathcal{D}^-\}$ with $F_l^- = \sum_k \gamma_{lk} C_k^- + \sum_{k'} \mu_{lk'} D_{k'}^-$. Also, let $\mathcal{F}^+ = \{F_1^+, F_2^+, \ldots, F_{|\mathcal{F}^-|}^+\}$, where

$$F_l^+ = F_l^- + (\sum_k \gamma_{lk} \beta_k) E, \;\; l \in \{1, 2, \ldots, |\mathcal{F}^-|\}. \tag{B.8}$$

  Note that $F_l^+ \in \text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\} \cap \text{span}\{C^+ \cup \mathcal{D}^+\}$.

  Suppose $x \in \text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\} \cap \text{span}\{C^+ \cup \mathcal{D}^+\}$, then there exists representations of x as follows.

$$x = \sum_k \eta_k A_k^- + \sum_{k'} \delta_{k'} B_{k'}^- \tag{B.9}$$

$$= \sum_l \xi_l (C_l^- + \beta_l E) + \sum_{l'} \zeta_{l'} D_{l'}^- \tag{B.10}$$

Therefore, we have

$$x - (\sum_l \xi_l \beta_l) E \in \text{span}\{\mathcal{A}^- \cup \mathcal{B}^-\} \cap \text{span}\{\mathcal{C}^- \cup \mathcal{D}^-\}$$

$$\Rightarrow x - (\sum_l \xi_l \beta_l) E = \sum_l \tau_l F_l^- = \sum_l \tau_l (F_l^+ - \sum_k \gamma_{lk} \beta_k) E)$$

Therefore,

$$x - \sum_l \tau_l F_l^+ = \Big(\sum_l \xi_l \beta_l - \sum_{k,l} \tau_l \gamma_{lk} \beta_k \Big) E = \Phi(x) E \tag{B.11}$$

We consider two cases here.

Sub-case 2a: First, suppose that $\Phi(x) = 0$ for all $x \in \text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\} \cap \text{span}\{\mathcal{C}^+ \cup \mathcal{D}^+\}$. Then, $\text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\} \cap \text{span}\{\mathcal{C}^+ \cup \mathcal{D}^+\} \subseteq \text{span}\{\mathcal{F}^+\}$. However, $\text{span}\{\mathcal{F}^+\} \subseteq \text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\} \cap \text{span}\{\mathcal{C}^+ \cup \mathcal{D}^+\}$. Hence, $\text{span}\{\mathcal{F}^+\} = \text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\} \cap \text{span}\{\mathcal{C}^+ \cup \mathcal{D}^+\}$. Next, we prove that members of $\mathcal{F}^+$ are linearly independent. Suppose $\sum_l \omega_l F_l^+ = 0$, then by (B.8),

$$\sum_l \omega_l F_l^- = \Big(\sum_{l,k} \omega_l \lambda_{lk} \beta_k \Big) E \tag{B.12}$$

Here, if $\mathcal{G}^- \neq \text{span}\{\mathcal{A}^-\}$, then $E \notin \mathcal{F}^-$ whp, and $\mathcal{F}^+$ are linearly independent, again whp. On the other hand, if $\mathcal{G}^- = \text{span}\{\mathcal{A}^-\}$, then $E \in \mathcal{F}^-$ can be uniquely represented as a linear combination of F_i^-, $i \in [|\mathcal{F}^-|]$. Let $E = \sum_l \psi_l F_l^-$. Given a particular value of $(\omega_1, \cdots, \omega_{|\mathcal{F}^-|}) \neq \mathbf{0}$, due to the randomness of the β_k's, the probability that $\sum_l \omega_l F_l^+ = 0$ happens is equal to $\frac{1}{q-1}$ which can be made as small as required by choosing a large field size.

Thus, $\mathcal{F}^+$ are linearly independent in this case. Therefore,

$$\begin{aligned} \dim(\text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\}) &= \dim(\text{span}\{F^+\}) \\ &= \dim(\text{span}\{F^-\}). \end{aligned}$$

Therefore, the update rule will be $b_S^+ = b_S^-$.

Sub-case 2b: suppose that $\Phi(x) \neq 0$ for some $x \in \text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\} \cap \text{span}\{\mathcal{C}^+ \cup \mathcal{D}^+\}$. Then, from (B.11), $E \in \text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\} \cap \text{span}\{\mathcal{C}^+ \cup \mathcal{D}^+\}$. Now, if $\mathcal{G}^- = \text{span}\{\mathcal{A}^-\}$, then $E \in \text{span}\{\mathcal{C}^- \cup \mathcal{D}^-\}$ which means that $\text{span}\{\mathcal{C}^+ \cup \mathcal{D}^+\} = \text{span}\{\mathcal{C}^- \cup \mathcal{D}^-\}$. Thus, the update rule in this case is given by $b_S^+ = b_S^-$. On the other hand, if $\mathcal{G}^- \neq \text{span}\{\mathcal{A}^-\}$, then $E \notin \text{span}\{\mathcal{C}^- \cup \mathcal{D}^-\}$. However, by (B.11), $E \in \text{span}\{\mathcal{C}^+ \cup \mathcal{D}^+\}$. Hence, there exists a representation of E as follows

$$E = \sum_l \pi_l (C_l^- + \beta_l E) + \sum_{l'} \varphi_{l'} D_{l'}^-. \tag{B.13}$$

This yields

$$\left(1 - \sum_l \pi_l \beta_l\right) E = \sum_l \pi_l C_l^- + \sum_{l'} \varphi_{l'} D_{l'}^-. \tag{B.14}$$

Given that $E \notin \text{span}\{\mathcal{C}^- \cup \mathcal{D}^-\}$, it follows from (B.14) that $\sum_l \pi_l \beta_l = 1$ which implies that

$$\sum_l \pi_l C_l^- + \sum_{l'} \varphi_{l'} D_{l'}^- = 0. \tag{B.15}$$

However, in Case 2, there cannot be an equation of the form (B.15), unless we have $\pi_l = 0$ for all l. Substituting $\pi_l = 0$ in (B.13) results in having $E \in \text{span}\{\mathcal{D}^-\}$. This is a contradiction, since $\mathcal{A}^-$ has innovative packets for $\mathcal{C}^- \cup \mathcal{D}^-$. Thus, the event $\Phi(x) \neq 0$ for some $x \in \text{span}\{\mathcal{A}^+ \cup \mathcal{B}^+\} \cap \text{span}\{\mathcal{C}^+ \cup \mathcal{D}^+\}$ occurs wlp. ■

B.4 Proof of Lemma 9.4

Here, we will use the complete definition of the occupancy vector, since we are considering the destination node. It is clear that if $S \subseteq [n]$ and $i \in S^c$, then $b_S^+ = b_S^-$.

Hence, we consider all $S \subseteq [n]$ where $i \in S$. Suppose packet E successfully transfers from relay i to d. Using the same argument as in the relay-to-relay case, if packet E is innovative to $S^c \cup \{d\}$ (i.e., $I_{\{i\} \to S^c} > 0$), successful transmission can increase $\dim(\text{span}\{S\} \cap \text{span}\{S^c \cup \{d\}\})$ by one because d always has enough space to store packets. Since $\dim(\text{span}\{S\})$ does not change by transmission of E, the update rule is $b_S^+ = b_S^- - 1$. Further, if packet E

is not innovative to $S^c \cup \{d\}$ (i.e., $I_{\{i\}\rightarrow S^c} = 0$), then either $E \in \text{span}\{d\}$ which results in no change in b_S, or $E \in \text{span}\{S^c\}$. The latter despite increasing $\dim(\text{span}\{S^c \cup \{d\}\})$ by at most one, will not change $\dim(\text{span}\{S\} \cap \text{span}\{S^c \cup \{d\}\})$ and hence does not alter b_S. ■

APPENDIX C

PROOFS OF RESULTS IN CHAPTER 10

C.1 Proof of Lemma 10.1

Suppose that a block of length K is decoded at time t^*. Then, at time t^*, the number of equations at the destination must have become equal to the number of unknowns, *i.e.*, $I_D(t^*) = P_D(t^*)$. Further, by definition, t_0 is the last time that the event $P_D(t) = I_D(t)$ has occurred before t^*, hence, $P_D(t) > I_D(t)$ for $t_0 < t < t^*$. Therefore, $I_D(t^*) - I_D(t_0)$ is the number of equations in the latest solvable set of linear equations, leading to find $I_D(t^*) - I_D(t_0)$ unknowns. The length of the decoded block being K, results in $I_D(t^*) - I_D(t_0) = K$. The proof of the reverse statement is straightforward and follows the same steps as mentioned.

C.2 Proof of Lemma 10.2

Given the channel realizations at time t, *i.e.*, whether if a packet is lost or not at time t, and knowing the way the innovativeness of each node changes with arrival and/or departure of packets, it is clear that $(I_S(t), I_R(t))$ only depends on $(I_S(t-1), I_R(t-1))$.

C.3 Proof of Lemma 10.3

Suppose that a stream of packets with source arrival rate λ is decodable. Assume that all of the states of the form $(0, Y)$ are transient, where $Y = 0, 1, \ldots, m$. In this case, after a certain amount of time and also after the last block of packet is decoded, the Markov chain $(I_S(t), I_R(t))$ will never visit any of the states of the form $(0, Y)$. Hence, at no point in time the number of equations generated and transmitted at the source will be as many as the number of unknowns used and hence, no block of packets will be ever decoded from that certain time forward. Next, suppose that in the Markov chain $(I_S(t), I_R(t))$, any state of the form $(0, Y)$ is recurrent, where $Y = 0, 1, \ldots, m$. Then, after visiting an arbitrary state $(0, i)$, the block of packets will be decoded with the successful transmission of i packets to the destination without receiving more packets from the source. Since i is finite ($i \leq m$), this event happens with a positive probability. Since a return to such states of the form $(0, Y)$ is recurrent, in a finite time the block of packets will be decoded.

C.4 Proof of Lemma 10.4

First, suppose all the states of the Markov chain $(I_S(t), I_R(t))$ are recurrent. Assuming an arbitrary state $I_S = i$ of the collapsed Markov chain $I_S(t)$ is transient means that there is a non-zero probability that $I_S(t)$ will never return to the state i and hence, the steady-state probability of state i is zero. Using Theorem 7.1, the sum of the steady-state probabilities of the group of states of the form (i, Y) in the Markov chain $(I_S(t), I_R(t))$ is equal to the steady-state probability of the state i in the collapsed Markov chain $I_S(t)$ which is zero, where $Y = 0, 1, \ldots, m$. This implies that the sum of the steady-state probabilities of the group of states of the form (i, Y) in the Markov chain $(I_S(t), I_R(t))$ is equal to zero which contradicts the assumption that all the states of the Markov chain $(I_S(t), I_R(t))$ are recurrent. Therefore, the initial assumption that an arbitrary state $I_S = i$ of the collapsed Markov chain $I_S(t)$ is transient must be false, and all the states of the collapsed Markov chain $I_S(t)$ are recurrent.

Next, suppose all the states of the collapsed Markov chain $I_S(t)$ are recurrent. Assuming an arbitrary state $(I_S, I_R) = (i, j)$ in the Markov chain $(I_S(t), I_R(t))$ is transient means that there is a non-zero probability that $(I_S(t), I_R(t))$ will never return to the state (i, j). Using Theorem 7.1. the sum of the steady-state probabilities of the group of states of the form (i, Y) in the Markov chain $(I_S(t), I_R(t))$ is equal to the steady-state probability of the state i in the collapsed Markov chain $I_S(t)$, where $Y = 0, 1, \ldots, m$. The steady-state probability of the state i in the collapsed Markov chain $I_S(t)$ is non-zero since all the states of the collapsed Markov chain $I_S(t)$ are recurrent. Hence, there is at least one state of the form (i, Y), in the Markov chain $(I_S(t), I_R(t))$ has a non-zero steady-state probability. Suppose that the state

(i, k) in the Markov chain $(I_S(t), I_R(t))$ has a non-zero steady state probability. Because of the structure of The the Markov chain $(I_S(t), I_R(t))$, the state (i, j) can be reached from the state (i, k) with a positive probability, *e.g.* by sending or receiving packets at the relay. Therefore, the state (i, j) is not transient and the result follows from this contradiction.

C.5 Proof of Lemma 10.5

Assumption **??** implies that the steady state probability $\pi_R(m)$ equals the blocking probability that the source S perceives from the relay node R and can be calculated from Equation (3.3) to be $\varepsilon_2 \Pr\{X_R = m\}$, where $\Pr\{X_R = m\}$ is the steady-state probability of the transmit-first Markov chain depicted in Figure 3.2. Since ε_2 is assumed to be a constant, to prove the lemma, we need to prove the results for $\Pr\{X_R = m\}$ instead of $\pi_R(m)$. In the transmit-first Markov chain depicted in Figure 3.2, $\alpha = r_{in}\varepsilon_2, \beta = \overline{r_{in}\varepsilon_2}$, and $\alpha_0 = r_{in}$, where r_{in} is the arrival rate of packets from the source. Clearly, r_{in} increases with λ because larger λ increases the probability of the source to be non-empty and hence increases the arrival rate of innovative packets to the relay from the sourse. Therefore, α and α_0 increase with λ and β decreases with λ. Intuitively, by examining the Markov chain in Figure 3.2, larger α and α_0, and smaller β leads to a larger steady-state probability for state m, *i.e.*, $\Pr\{X_R = m\}$. Hence, $\Pr\{X_R = m\}$ is a non-decreasing function of λ and from Equation (3.2), it can be seen that it is a continuous function as well. Further, by increasing λ the probability of the source being in empty state decreases. However, at some point increasing λ leads to a situation in which the empty state of the source becomes transient or null-recurrent. In this case, the parameters α, α_0 and β will not change anymore and $\pi_R(m)$ achieves its maximum $\pi_R^{max}(m)$.

C.6 Proof of Theorem 10.1

Using Lemma 10.3 and Lemma 10.4, it is clear that to prove the theorem we need to prove that the state 0 in the collapsed Markov chain $I_S(t)$ is recurrent if and only if $\lambda < C^*$, where

$C^* = \overline{\varepsilon}_1 \overline{\pi}_R^{max}(m)$.

Suppose that former holds, *i.e.*, the state 0 in the Markov chain $I_S(t)$ is recurrent. Then, assume that $\lambda \geq C^*$. Further, let $r_{out}(\lambda)$ be the maximum possible departure rate at the source which equals $\overline{\varepsilon}_1 \overline{\pi}_R(m)$. From Lemma 10.5, we know that $\pi_R(m)$ is a non-decreasing continuous function of λ, achieving its maximum, $\pi_R^{max}(m)$, when all the states in the collapsed Markov chain $I_S(t)$ are transient or null-recurrent. Hence, $r_{out}(\lambda)$ is a non-increasing continuous function of λ, achieving its minimum, $r_{out}^{min} = \overline{\varepsilon}_1 \overline{\pi}_R^{max}(m) = C^*$, when all the states in the collapsed Markov chain $I_S(t)$ are transient or null-recurrent. Since the state 0 in the Markov chain $I_S(t)$ is recurrent, it is clear that the arrival rate at the source is smaller than the maximum possible departure rate, *i.e.*, $\lambda < r_{out}(\lambda)$. It is also known that $r_{out}(\lambda) \geq C^*$ since $C^* = r_{out}^{min}$. Let λ^* be the smallest arrival rate at the source for which the state 0 of the Markov chain $I_S(t)$ is transient or null-recurrent meaning for any arrival rate smaller than λ^* the state 0 is recurrent, *i.e.*, $\lambda < r_{out}(\lambda)$ for any $\lambda < \lambda^*$. Then, because $r_{out}(\lambda)$ is a continuous function of λ, we have $\lambda^* = r_{out}(\lambda^*)$. Further, $r_{out}(\lambda^*) = r_{out}^{min} = C^*$ because $r_{out}(\cdot)$ achieves its minimum when all the states in the Markov chain $I_S(t)$ are transient or null-recurrent. Note that, if state 0 is transient, then every other state in $I_S(t)$ is also transient. Hence, we have $\lambda^* = C^*$ and consequently, $\lambda < C^*$ which is a contradiction to the assumption $\lambda \geq C^*$. Therefore, the assumption $\lambda \geq C^*$ must be false which proves the results.

The proof of the reverse is straightforward. Assuming $\lambda < C^*$ guarantees that the state 0 in the Markov chain $I_S(t)$ is recurrent since $C^* = \overline{\varepsilon}_1 \overline{\pi}_R^{max}(m)$ is the minimum of the maximum possible departure rates at the source and hence guarantees that the arrival rate λ is smaller than any maximum departure rates at the source.

C.7 Proof of Lemma 10.7

First, we need to find the condition for decoding the packets arrived at the source up to time t_0. Right after $I_S(t)$ becomes zero, all the needed useful equations for the destination to decode the packets arrived at the source up to time t_0 are now stored at the relay node. Further, $I_R(t_0) = k$ implies that there are only k of such equations available at the relay node. Therefore, to be able to decode, the relay node should not receive any innovative packet from the source while the destination is receiving k packets from the relay. Let δ be the probability of the event that in a single time epoch source transmits a packet and the packet is either lost or not innovative for the relay. Since the source is empty at t_0, there is a higher chance that the source remains empty at the next few epochs, leading to $\delta = 1$. However, after a few epochs, a packet arrives at the source and we have $\delta = \varepsilon_1$. Hence, assuming $\delta \geq \varepsilon_1$ is a reasonable approximation for the purpose of steady-state analysis. Consider the scenario in which the task of decoding will be completed in exactly $k + i$ epochs[1], where $i = 0, 1, 2, \ldots$. We proceed to compute the probability of this scenario. In i of the epochs from the first $i + k - 1$ epochs, at the relay, neither an innovative packet should be received nor a packet should be successfully transmitted, which happens with probability $\delta\varepsilon_2$ in a single epoch. Further, in $k - 1$ of the epochs from the first $i + k - 1$ epochs, At the relay, a packet has to be successfully transmitted to the destination while no packet arrives from the source, which happens with probability $\delta\overline{\varepsilon}_2$ in a single epoch. Finally, in the last epoch, a packet has to be received by the destination, which happens with probability $\overline{\varepsilon}_2$. Therefore, we have the following:

$$
\begin{aligned}
P_R^{dec}(k) &= \sum_{i=0}^{\infty} \binom{k+i-1}{i} \{\delta\varepsilon_2\}^i \{\delta\overline{\varepsilon}_2\}^{k-1}\overline{\varepsilon}_2 && \text{(C.1)}\\
&\geq \sum_{i=0}^{\infty} \binom{k+i-1}{i} \{\varepsilon_1\varepsilon_2\}^i \{\varepsilon_1\overline{\varepsilon}_2\}^{k-1}\overline{\varepsilon}_2 && \text{(C.2)}\\
&= \{\varepsilon_1\overline{\varepsilon}_2\}^{k-1}\overline{\varepsilon}_2 \sum_{i=0}^{\infty} \frac{(k+i-1)\cdots(k)}{i!}\{\varepsilon_1\varepsilon_2\}^i && \text{(C.3)}\\
&> \{\varepsilon_1\overline{\varepsilon}_2\}^{k-1}\overline{\varepsilon}_2 \sum_{i=0}^{\infty} \frac{k^i}{i!}\{\varepsilon_1\varepsilon_2\}^i && \text{(C.4)}\\
&= \{\varepsilon_1\overline{\varepsilon}_2\}^{k-1}\overline{\varepsilon}_2 e^{k\varepsilon_1\varepsilon_2} && \text{(C.5)}
\end{aligned}
$$

Note that, (C.2) is the result of assuming $\delta \geq \varepsilon_1$.

[1]The minimum number of epochs to complete the decoding is k.

C.8 Proof of Theorem 10.2

Before any block of packets is decoded at the destination, the following events must occur: $I_S(t')$ returns to the state 0, and $I_R(t') = k$ with probability $\pi_R^{rcv}(k)$, where $k = 1, 2, \ldots, m$. For each k, all the packets arrived at the source up to time t' will be decoded with probability $P_R^{dec}(k)$. Therefore, every time $I_S(t)$ returns to zero at epoch t', all the packets arrived at the source up to time t' will be decoded with the average probability $\sum_{k=1}^{m} \pi_R^{rcv}(k) P_R^{dec}(k)$. Further, since the expected waiting time for $I_S(t)$ to return to zero is $E\left[T_0^+\right]$, the average time it takes for a block of packets to be decoded at the destination is $E\left[T_0^+\right]\left\{\sum_{k=1}^{m} \pi_R^{rcv}(k) P_R^{dec}(k)\right\}^{-1}$. Finally, the rate at which the destination receives innovative packets is λ given that the Markov chain $I_S(t)$ is ergodic, which is the case since we assume the stream is decodable. Hence, $\lambda E\left[T_0^+\right]\left\{\sum_{k=1}^{m} \pi_R^{rcv}(k) P_R^{dec}(k)\right\}^{-1}$ will be the average length of a decoded block, and the results follows.

C.9 Proof of Lemma 10.8

The proof is very similar to the proof of Lemma 10.7. In Lemma 10.7, $\bar{\varepsilon}_2$ represents the probability that a packet is successfully transmitted from the relay to the destination. However, here, if a relay v_j is transmitting a packet to v_{j+1}, it would count as successful only when the packet is not lost, which occurs with probability $\bar{\varepsilon}_{j+1}$, and also the relay v_{j+1} is not full, which occurs with probability $\bar{\pi}_{j+1}(m_{j+1})$. Therefore, the probability that a packet is successfully transferred from v_j to v_{j+1} is $r_{j+1} = \bar{\varepsilon}_{j+1}\bar{\pi}_{j+1}(m_{j+1})$

C.10 Proof of Theorem 10.3

A block of packets is decoded at the destination when the following events occur: $I_S(t')$ returns to the state 0, and $I_1(t') = k_1$ with probability $\pi_1^{rcv}(k_1)$, where $k_1 = 1, 2, \ldots, m_1$. For each k_1, all the information required to decode the original packets arrived at the source up to time t' will be passed to v_1 with a probability bounded above by $P_1^{dec}(k_1)$. Therefore, every time $I_S(t)$ returns to zero at epoch t', all the information required to decode the original packets arrived at the source up to time t' will be passed to v_1 with an average probability bounded above by $\sum_{k_1=1}^{m_1} \pi_1^{rcv}(k_1) P_1^{dec}(k_1)$. Similarly, all the information required to decode only the original packets arrived at the source up to time t' will be passed to v_2 with the average probability bounded above by $\sum_{k_2=1}^{m_2} \pi_2^{rcv}(k_2) P_2^{dec}(k_2)$, and so on. Finally, all the packets arrived at the source up to time t' will be decoded with an average probability bounded above by $\sum_{k_{h-1}=1}^{m_{he1}} \pi_{h-1}^{rcv}(k_{h-1}) P_1^{dec}(k_1)$. Further, since the expected waiting time for $I_S(t)$ to return to zero is $E\left[T_0^+\right]$, the average time it takes for a block of packets to be decoded at the destination is $E\left[T_0^+\right] \prod_{j=1}^{h-1} \left\{ \sum_{k=1}^{m_j} \pi_j^{rcv}(k) P_j^{dec}(k) \right\}^{-1}$. The rest of the proof is similar to the proof of Theorem 10.2.

REFERENCES

[1] C. Villamizar and C. Song, “High performance TCP in ANSNet,” *ACM SIGCOMM Computer Communication Review*, vol. 24, no. 5, pp. 45–60, 1994.

[2] G. Appenzeller, I. Keslassy, and N. McKeown, “Sizing router buffers,” *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 281–292, 2004.

[3] M. E. Yashar, M. Enachescu, Y. Ganjali, A. Goel, N. Mckeown, and T. Roughgarden, “Routers with very small buffers,” in *IEEE Infocom*, 2006.

[4] G. Appenzeller, “Sizing router buffers,” *PhD Thesis, Dept. of EE, Stanford University*, 2005.

[5] K. Avrachenkov, U. Ayesta, and A. Piunovskiy, “Optimal choice of the buffer size in the internet routers,” *Proceedings of IEEE Conference on Decision and Control*, 2005.

[6] L. Andrew and et al., “Buffer sizing for nonhomogeneous TCP sources,” *IEEE Communications Letters*, vol. 9, no. 6, pp. 567–569, 2005.

[7] D. Y. Eun and X. Wang, “Achieving 100 throughput in TCP/AQM under aggressive packet marking with small buffer,” *IEEE/ACM Transactions on Networking*, vol. 16, no. 4, pp. 945–956, 2008.

[8] G. Raina and D. Wischik, “Buffer sizes for large multiplexers: TCP queueing theory and instability analysis,” *Proceedings of EuroNGI*, 2005.

[9] D. Wischik, “Buffer requirements for high-speed routers,” *Proceedings of ECOC*, 2005.

[10] D. Wischik and N. McKeown, “Part I: Buffer sizes for core routers,” *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 2, pp. 75–78, 2005.

[11] G. Raina, D. Towsley, and D. Wischik, “Part II: Control theory for buffer sizing,” *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 2, pp. 79–82, 2005.

[12] M. Enachescu and et al., “Part III: Routers with very small buffers,” *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 2, pp. 83–89, 2005.

[13] M. Enachescu and et al., “Routers with very small buffers,” *Proceedings of IEEE INFOCOM*, 2006.

[14] N. Beheshti and et al., “Buffer sizing in all-optical packet switches,” *Proceedings of OFC/NFOEC*, 2006.

[15] N. Beheshti and et al., “Obtaining high throughput in networks with tiny buffers,” *Proceedings of IEEE IWQoS*, 2008.

[16] S. Gorinsky, A. Kantawala, and J. Turner, “Link buffer sizing: A new look at the old problem,” *Proceedings of ISCC*, 2005.

[17] S. Gorinsky, A. Kantawala, and J. Turner, “Simulation perspectives on link buffer sizing,” *Simulation*, vol. 83, no. 3, pp. 245–257, 2007.

[18] B. Zhao, A. Vishwanath, and V. Sivaraman, “Performance of high-speed TCP applications in networks with very small buffers,” *Proceedings of IEEE ANTS*, 2007.

[19] A. Razdan and et al., “Enhancing TCP performance in networks with small buffers,” *Proceedings of IEEE ICCCN*, 2002.

[20] O. Alparslan, S. Arakawa, and M. Murata, “Performance of paced and non-paced transmission control algorithms in small buffered networks,” *Proceedings of ISCC*, 2006.

[21] A. Lakshmikantha and et al., "Buffer sizing results for RCP congestion control under file arrivals and departures," accepted in *ACM SIGCOMM Computer Communication Review, 2009*.

[22] J. Sommers and et al., "An SLA perspective on the router buffer sizing problem," *ACM SIGMETRICS Performance Evaluation Review*, vol. 35, no. 4, pp. 40–51, 2008.

[23] N. Beheshti and et al, "Experimental study of router buffer sizing," *ACM/USENIX IMC*, 2008.

[24] R. G. Gallager, *Information theory and reliable communications*. John Wiley and Sons, New York, 1968.

[25] M. Mushkin and I. Bar-David, "Capacity and coding for the Gilbert-Elliott channels," *IEEE Trans. on Inform. Theory*, vol. 35, pp. 1277–1290, November 1989.

[26] S. N. Diggavi and M. Grossglauser, "Information transmission over a finite buffer channel," *IEEE Trans. on Inform. Theory*, vol. 52, no. 3, pp. 1226–1237, 2006.

[27] P. Pakzad, C. Fragouli, and A. Shokrollahi, "Coding schemes for line networks," *IEEE International Symposium on Inform. Theory*, Sept. 2005.

[28] D. S. Lun, P. Pakzad, C. Fragouli, M. Medard, and R. Koetter, "An analysis of finite-memory random linear coding on packet streams," *in Proc. of the 2nd Workshop on Network Coding, Theory, and Applications (NetCod 2006), Boston, MA, April 3-7, 2006*.

[29] B. N. Vellambi, N. Rahnavard, and F. Fekri, "The effect of finite memory on throughput of wireline packet networks," *Information Theory Workshop (ITW 2007), Lake Tahoe, CA, Sept. 2–6, 2007*.

[30] B. N. Vellambi and F. Fekri, “On the throughput of acyclic wired packet networks with finite buffers,” *Information Theory and Applications Workshop (ITA)*, 2008.

[31] B. Vellambi, N. Torabkhani, and F. Fekri, “Throughput and latency in finite-buffer line networks,” *IEEE Transactions on Information Theory*, vol. 57, pp. 3622–3643, June 2011.

[32] M. A. Shokrollahi, “Capacity-achieving sequences,” *in IMA Volumes in Mathematics and its Applications*, vol. 123, pp. 153–166, 2000.

[33] P. Oswald and M. A. Shokrollahi, “Capacity-achieving sequences for the erasure channel,” *IEEE Trans. Inform. Theory*, vol. 48, pp. 3017–3028, December 2002.

[34] T. J. Richardson and R. L. Urbanke, “The capacity of low-density parity-check codes under message-passing decoding,” *IEEE Trans. Inform. Theory*, vol. 47, pp. 599–618, February 2001.

[35] P. Maymounkov and D. Mazieres, “Rateless codes and big downloads,” *Proc. of the 2nd International Workshop on Peer-toPeer Systems*, 2003.

[36] P. Maymounkov, “Online codes,” *NYU Technical Report TR2003-883*, 2002.

[37] M. Luby, “LT codes,” *43rd Annual IEEE Symposium on Foundations of Computer Science*, 2002.

[38] M. Mitzenmacher, “Digital fountains: A survey and look forward,” *Information Theory workshop*, pp. 271–276, Oct. 2004.

[39] A. Shokrollahi, “Raptor codes,” *IEEE Transactions on Information Theory*, vol. 52, pp. 2551–2567, 2006.

[40] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, “Network information flow,” *IEEE Trans. on Inform. Theory*, vol. 46, pp. 1204–1216, July 2000.

[41] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. on Inform. Theory*, vol. 49, pp. 371–381, February 2003.

[42] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Trans. Networking*, vol. 11, pp. 782–795, October 2003.

[43] D. S. Lun, N. Ratnakar, M. Medard, R. Koetter, D. R. Karger, T. Ho, E. Ahmed, and F. Zhao, "Minimum-cost multicast over coded packet networks," *IEEE Trans. Inform. Theory*, vol. 52, pp. 2608–2623, June 2006.

[44] D. S. Lun, M. Medard, and M. Effros, "On coding for reliable communication over packet networks," *in Proc. of Allerton conference on communication, control, and computing, Monticello, IL, 2004*.

[45] Y. Wu, P. A. Chou, and S.-Y. Kung, "Minimum-energy multicast in mobile ad hoc networks using network coding," *IEEE Trans. on Communications*, vol. 53, pp. 1906–1918, November 2005.

[46] D. S. Lun, N. Ratnakar, R. Koetter, M. Medard, E. Ahmed, and H. Lee, "Achieving minimum-cost multicast: A decentralized approach based on network coding," *24th Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM*, vol. 3, pp. 1607–1617, March 2005.

[47] D. S. Lun, M. Medard, and R. Koetter, "Efficient operation of wireless packets networks using network coding," *International Workshop on Convergent Technology*, 2005.

[48] D. S. Lun, M. Medard, T. Ho, and R. Koetter, "Network coding with a cost criterion," Intern. Symp. on Info. Theory and its Appl.(ISITA 2004), Oct. 2004.

[49] A. F. Dana, R. Gowaikar, R. Palanki, B. Hassibi, and M. Effros, “Capacity of wireless erasure networks.,” *IEEE Transactions on Information Theory*, vol. 52, no. 3, pp. 789–804, 2006.

[50] A. F. Dana, R. Gowaikar, and B. Hassibi, “On the capacity region of broadcast over wireless erasure networks,” in *Forty-Second Annual Allerton Conference on Communication, Control and Computing*, October 2004.

[51] R. Koetter and M. Médard, “An algebraic approach to network coding,” *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, 2003.

[52] S.-Y. R. Li, R. W. Yeung, and N. Cai, “Linear network coding,” *IEEE Transactions on Information Theory*, vol. 49, pp. 371–381, February 2003.

[53] T. Altiok, “Approximate analysis of queues in series with phase-type service times and blocking,” *Operations Research*, vol. 37, pp. 301–310, July 1989.

[54] T. Altiok, “Approximate analysis of exponential tandem queues with blocking,” *European Journal of Operational Research*, vol. 11, no. 4, pp. 390–398, 1982.

[55] A. Brandwajn and Y.-L. L. Jow, “An approximation method for tandem queues with blocking,” *Operations Research*, vol. 36, no. 1, pp. 73–83, 1988.

[56] K. C. So and K. tsai E. Chin, “Performance bounds on multiserver exponential tandem queues with finite buffers,” *European Journal of Operational Research*, vol. 63, no. 3, pp. 463–477, 1992.

[57] Y. Dallery and Y. Frein, “On decomposition methods for tandem queueing networks with blocking,” *Operations Research*, vol. 41, no. 2, pp. 386–399, 1993.

[58] R. Serfozo, *Introduction to Stochastic Networks*. New York: Springer-Verlag, 1st ed., 1999.

[59] H. G. Perros, *Queueing networks with blocking.* New York, NY, USA: Oxford University Press, Inc., 1994.

[60] L. Tassiulas and A. Ephremides, “Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks,” *IEEE Transactions on Automatic Control*, vol. 37, pp. 1936 –1948, Dec. 1992.

[61] P. Giaccone, E. Leonardi, and D. Shah, “Throughput region of finite-buffered networks,” *IEEE Trans. on Parallel and Distributed Systems*, vol. 18, no. 2, pp. 251–263, 2007.

[62] L. B. Le, E. Modiano, and N. Shroff, “Optimal control of wireless networks with finite buffers,” *Proceedings of IEEE INFOCOM*, March 2010.

[63] M. J. Neely and R. Urgaonkar, “Optimal backpressure routing for wireless networks with multi-receiver diversity,” *Ad Hoc Networks*, vol. 7, no. 5, pp. 862–881, 2009.

[64] M. Neely, E. Modiano, and C.-P. Li, “Fairness and optimal stochastic control for heterogeneous networks,” *IEEE/ACM Transactions on Networking*, vol. 16, pp. 396 –409, April 2008.

[65] M. Neely, E. Modiano, and C. Rohrs, “Dynamic power allocation and routing for time-varying wireless networks,” *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 89 – 103, Jan. 2005.

[66] P. Chaporkar, K. Kar, X. Luo, and S. Sarkar, “Throughput and fairness guarantees through maximal scheduling in wireless networks,” *IEEE Transactions on Information Theory*, vol. 54, pp. 572 –594, feb. 2008.

[67] L. Huang, S. Moeller, M. Neely, and B. Krishnamachari, “Lifo-backpressure achieves near optimal utility-delay tradeoff,” in *IEEE International Symposium on*

Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), pp. 70–77, May 2011.

[68] Y. E.Sagduyu and A. Ephremides, “Network coding in wireless queueing networks: Tandem network case,” *2006 IEEE International Symposium on Information Theory*, pp. 192–196, July 2006.

[69] Y. E. Sagduyu and A. Ephremides, “On network coding for stable multicast communication,” *IEEE Military Communications Conference MILCOM*, October 2007.

[70] J. M. Smith, “Properties and performance modelling of finite buffer M/G/1/K networks,” *Computers and Operations Research*, vol. 38, no. 4, pp. 740–754, 2011. Available online 1 September 2010.

[71] D. Aldous and J. Fill, “Reversible markov chains and random walks on graph,” *Monograph in preparation, available at http://www.stat.berkeley.edu/users/aldous/RWG/book.html*, 2002.

[72] N. Torabkhani, B. N. Vellambi, and F. Fekri, “Throughput and latency of acyclic erasure networks with feedback in a finite buffer regime,” *in Proc. of IEEE Information Theory Workshop (ITW 2010), Dublin, Ireland*, Aug. 2010. Also available at `http://arxiv.org/abs/1012.2621`.

[73] R. Chandra and P. Bahl, “MultiNet: connecting to multiple IEEE 802.11 networks using a single wireless card,” in *Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, vol. 2, pp. 882–893, March 2004.

[74] J. MACGREGOR SMITH and F. R. B. CRUZ, “The buffer allocation problem for general finite buffer queueing networks,” *IIE Transactions*, vol. 37, no. 4, pp. 343–365, 2005.

[75] V. Anantharam, “The optimal buffer allocation problem,” *IEEE Transactions on Information Theory*, vol. 35, pp. 721–725, July 1989.

[76] P. Giaccone, E. Leonardi, and D. Shah, “On the maximal throughput of networks with finite buffers and its application to buffered crossbars,” in *Proc. 24th Annual Joint Conference of the IEEE Computer and Communications Societies(INFOCOM).*, vol. 2, pp. 971–980, March 2005.

[77] K. Jamshaid, B. Shihada, L. Xia, and P. Levis, “Buffer sizing in 802.11 wireless mesh networks,” in *IEEE 8th International Conference on Mobile Adhoc and Sensor Systems (MASS)*, pp. 272–281, October 2011.

[78] D. Malone, P. Clifford, and D. Leith, “On buffer sizing for voice in 802.11 WLANs,” *IEEE Communications Letters*, vol. 10, pp. 701–703, Oct. 2006.

[79] T. Li, D. Leith, and D. Malone, “Buffer sizing for 802.11-based networks,” *IEEE/ACM Transactions on Networking*, vol. 19, pp. 156–169, Oct. 2011.

[80] L. E. Meester and J. G. Shanthikumar, “Concavity of the throughput of tandem queueing systems with finite buffer storage space,” in *Advances in Applied Probability*, vol. 22, pp. 764–767, September 1990.

[81] S. Merugu, M. Ammar, and E. Zegura, “Routing in space and time in networks with predictable mobility,” *Technical Report GIT-CC-04-7, Georgia Institute of Technology.*, 2004.

[82] L.-J. Chen, C.-H. Yu, T. Sun, Y.-C. Chen, and H. hua Chu, “A hybrid routing approach for opportunistic networks,” *CHANTS ’06: Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, pp. 213–220, 2006.

[83] Y. Lin, B. Li, and B. Liang, “Stochastic analysis of network coding in epidemic routing,” *IEEE Journal on Selected Areas in Communications*, vol. 26, pp. 794–808, June 2008.

[84] Y. Lin, B. Liang, and B. Li, “Performance modeling of network coding in epidemic routing,” in *Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, MobiOpp ’07, pp. 67–74, 2007.

[85] R. Groenevelt, P. Nain, and G. Koole, “The message delay in mobile ad hoc networks,” *Perform. Eval.*, vol. 62, no. 1-4, pp. 210–228, 2005.

[86] A. Al-Hanbali, A. A. Kherani, and P. Nain, “Simple models for performance evaluation of a class of two-hop relay protocols,” *in Proceedings of the IFIP Conference on Networking*, May 2007.

[87] X. Zhang, G. Neglia, J. Kurose, and D. Towsley, “Performance modeling of epidemic routing,” *Comp. Net.*, vol. 51, no. 10, pp. 2867–2891, 2007.

[88] T. Spyropoulos, K. Psounis, and C. Raghavendra, “Efficient routing in intermittently connected mobile networks: The single-copy case,” *ACM/IEEE Trans. on Networking*, vol. 16, pp. 63–76, February 2008.

[89] R. Subramanian, B. Vellambi, and F. Fekri, “A generalized framework for throughput analysis in sparse mobile networks,” in *Proceedings of the 7th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pp. 1–10, June 2009.

[90] R. Subramanian and F. Fekri, “Analysis of multiple-unicast throughput in finite-buffer delay-tolerant networks,” in *IEEE International Symposium on Information Theory (ISIT)*, pp. 1634–1638, July 2009.

[91] R. Subramanian and F. Fekri, "Throughput performance of network-coded multicast in an intermittently-connected network," in *Proc. of the 8th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, pp. 212–221, June 2010.

[92] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Trans. Netw.*, vol. 2, pp. 1–15, February 1994.

[93] V. Paxson and S. Floyd, "Wide area traffic: the failure of poisson modeling," *IEEE/ACM Transactions on Networking*, vol. 3, pp. 226–244, June 1995.

[94] W.-H. Zhou and A.-H. Wang, "Discrete-time queue with bernoulli bursty source arrival and generally distributed service times," *Applied Mathematical Modelling*, vol. 32, no. 11, pp. 2233–2240, 2008.

[95] N. Torabkhani, B. N. Vellambi, A. Beirami, and F. Fekri, "Exact modeling of the performance of random linear network coding in finite-buffer networks," in *Proc. of IEEE Information Theory Workshop (ITW), Paraty, Brazil*, pp. 538–542, Oct. 2011. Also available at `http://arxiv.org/abs/1112.2810`.

[96] T. Ho, M. Medard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory,*, vol. 52, Oct. 2006.

[97] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Trans. on Networking*, vol. 11, pp. 782–795, oct. 2003.

[98] P. Chou, Y. Wu, and K. Jain, "Practical network coding," *Proc. 41st Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL*, oct. 2003.

[99] D. S. Lun, M. Mdard, R. Koetter, and M. Effros, "On coding for reliable communication over packet networks," *Physical Communication*, vol. 1, no. 1, pp. 3–20, 2008.

[100] W. Feller, *An introduction to probability theory and its applications (2 vols)*. John Wiley & Sons, 2 ed., 1957. vol 1 missing: alkan.